Taming the Duplication-Loss-Coalescence Model with Integer Linear Programming

Jarosław Paszek*

Faculty of Mathematics, Informatics and Mechanics

University of Warsaw, Poland

E-mail: jpaszek@mimuw.edu.pl

Alexey Markin*

Department of Computer Science

Iowa State University, USA

E-mail: amarkin@iastate.edu

Paweł Górecki

Faculty of Mathematics, Informatics and Mechanics

University of Warsaw, Poland

E-mail: gorecki@mimuw.edu.pl

Oliver Eulenstein

Department of Computer Science

Iowa State University, USA

E-mail: oeulenst@iastate.edu

(*) Corresponding authors. Both authors contributed equally.

Key words: Phylogenetics, Duplications, Losses, Coalescence, Reconciliation, ILP, DLC

Abstract

The Duplication-Loss-Coalescence parsimony model (DLC-model) is invaluable for analyzing the complex scenarios of concurrent duplication-loss and deep coalescence events in the evolution of gene families. However, inferring such scenarios for already moderately-sized families is prohibitive due to the computational complexity involved. To overcome this stringent limitation, we make the first step by describing a flexible Integer linear programming (ILP) formulation for inferring DLC evolutionary scenarios. Then, to make the DLC-model more scalable, we introduce four sensibly constrained versions of the model and describe modified versions of our ILP formulation reflecting these constraints. Our simulation studies showcase that our constrained ILP formulations compute evolutionary scenarios that are substantially larger than scenarios computable under our original ILP formulation and the original dynamic programming algorithm by Wu et al. Further, scenarios computed under our constrained DLC-models are remarkably accurate compared to corresponding scenarios under the original DLC-model, which we also confirm in an empirical study with thousands of gene families.

1 Introduction

Reconstructing gene families' evolutionary histories, referred to as gene trees, is central to understanding gene and protein function. Gene trees make comparative and investigative studies possible that illuminate relationships between the structure and function among orthologous groups of genes and are an indispensable tool for assessing the functional diversity and specificness of biological interlinkage for genes within the same family (Ohno, 1970; Koonin, 2005; Lynch and Conery, 2000; Page, 1994; Arvestad *et al.*, 2004).

Crucial for understanding evolutionary histories of gene families is contemplating them against a respective species phylogeny, i.e., the evolutionary history of species that host(ed) the genes under consideration. This approach is known as *gene tree reconciliation*, and it can directly reveal the most valuable evolutionary events of interest, such as (i) *gene duplication*, (ii) *gene loss*, and (iii) *deep coalescence* or *incomplete lineage-sorting* (which appear as a result of a genetic polymorphism surviving speciation).

Traditional tree reconciliation approaches, while computationally efficient, are rather limited in practice, as they either only account for duplication and loss events or, on the other hand, only for deep coalescence events (Wu and Zhang, 2011; Górecki and Tiuryn, 2006; Maddison, 1997). Beyond the traditional approaches, recently, a robust unified *duplication-loss-coalescence (DLC)* approach has been developed that simultaneously accounts for duplications, losses, and deep coalescence events.

In particular, Rasmussen and Kellis, 2012 originally developed a rigorous statistical model referred to as *DLCoal*. Then, a computationally more feasible parsimony framework, which we refer to here as *DLC-model*, was developed by Wu *et al.*, 2014. The DLC-model is a discrete version of the DLCoal model. It was shown to be very effective in identifying ortholog/paralog relations and accurate inference of the duplication loss events. Wu et al. additionally presented an optimized strategy for enumerating possible reconciliation scenarios and a dynamic programming solution to find the optimum reconciliation cost; this algorithm is implemented in the "DLCPar" software package (Wu *et al.*, 2014). For convenience, from

now on we refer to their implementation as *DP* (dynamic programming).

While it has been demonstrated that DLC-model is computationally more feasible when compared to DLCoal, DP is still only applicable to reconciliation problems involving less than 200 genes. Limiting evolutionary studies to such a small number of genes is highly restrictive in practice, where frequently gene families with thousands of genes and hundreds of host species appear (Li *et al.*, 2006). Further, DP is not scalable due to its exponential runtime (Du *et al.*, 2019a). Naturally, there is a demand for novel models that are (i) efficiently computable and (ii) comparable to DP in terms of accuracy.

This work presents a non-trivial and flexible *integer linear programming (ILP)* formulation of the DLC-model optimization problem. We formulate four novel and constrained DLC-models for improved scalability and use our ILP formulation to validate these constrained models. That is, our models have smaller solution space and, therefore, are more efficiently computable than the original DLC-model (see "Our contribution" for more details).

Related work. In recent years, there has been an increased interest in phylogenetic methods involving simultaneous modeling of duplication, loss, and deep coalescence events (Szöllősi *et al.*, 2014; Du *et al.*, 2019c). For example, recently, an approach for *co-estimation* of the gene trees and the respective species tree based on the DLCoal model was presented (Du and Nakhleh, 2018). Further, Chan *et al.*, 2017 showed a parsimony framework for the reconciliation of a gene tree with a species tree by simultaneously modeling DLC events as well as horizontal gene transfer events. While promising, their approach remains computationally challenging.

Recently, Carothers *et al.*, 2020 have independently developed an ILP formulation for the DLC-model that overlaps with our preceding work (Ansarifar *et al.*, 2020). Notably, their formulation showed comparable results with DP in experiments.

Our contribution. We developed a flexible ILP formulation that solves DLC-model. During this formulation development, we observed formal issues with the original definition of the DLC-model (Wu *et al.*, 2014). Consequently, we present corrected and improved model

definitions equivalent to the original model definitions in this work. Especially, we corrected problems with the definition of a partial order on gene tree nodes, which could otherwise lead to incorrect scoring of deep coalescence events (see Section 2 for the full updated model definitions).

Further, we investigated novel approaches based on the original DLC-model yielding more computationally feasible but still biologically relevant models. In particular, we observed that the advanced time complexity of DP originates from allowing the duplications to appear at any edge of the gene tree, even if the data may not support such occurrences. While this flexibility allows accounting for all potentially possible DLC scenarios, we show that constraining the duplication locations to those with direct evidence of duplications will dramatically improve the efficiency of computing optimum reconciliations (without losing the accuracy).

The second major cause of the DLC-model's computational complexity is the ordering of gene tree nodes within the same locus. While such ordering is crucial for proper counting of the incomplete lineage sorting events, the accurate order of gene tree nodes can be often inferred using the molecular clock methods (Suchard *et al.*, 2018; Sagulenko *et al.*, 2018). Therefore, we can eliminate the computational complexity that stems from the enumeration of all possible gene tree orderings currently required by the DLC-model.

Consequently, to advance the applicability of DLC-model, we propose four additional models and the respective (additionally constrained) ILP formulations based on the above observations (see Section 2.4).

To evaluate the performance of the ILPs and test our constrained models, we designed a comprehensive simulation study with a range of parameters derived from the 16 fungi dataset (Rasmussen and Kellis, 2012), which became a standard for duplication-loss-coalescence simulations (Molloy and Warnow, 2019; Du *et al.*, 2019b; Wu *et al.*, 2014). As a result, we observed that the unconstrained ILP is generally more scalable than DP and can manage complex instances, which were infeasible for DP by Wu et al. (see Section 4 for more de-

tails). Further, most of the constrained ILPs proved to be efficient even on larger datasets with more than 200 genes, where DP and the unconstrained ILP failed. Out of the three highly efficient ILPs, two were also remarkably accurate. In particular, they were accurate for 98.3% of datasets, while reducing the computational time significantly.

Next, we evaluated our ILP formulations on a baseline empirical dataset with $5,351$ gene families in fungi (Butler *et al.*, 2009). This study demonstrated the practical advantage of our proposed model constraints. In particular, fixing coalescence orders using the strict molecular clock assumption proved to be highly effective. In combination with duplication constraints, this strategy then yields a highly efficient method. Crucially, we additionally developed a novel efficient rooting strategy for weighted gene trees using the classic duplication-loss model (Page, 1994) and demonstrated its applicability on this dataset.

Finally, we note that an essential advantage of using ILPs is that one can terminate an ILP solver early but still achieve a good approximation of the optimum reconciliation cost due to the intricate optimization algorithms used by ILP solvers.

2 Methods

In this section, first, we introduce a largely revised and corrected DLC-model definitions and other preliminaries. Secondly, we describe our core (unconstrained) ILP formulation. Next, we introduce the biologically-significant constraints to DLC-model and the respective constrained ILP formulations. Finally, we conclude with a discussion of our ILP formulations' complexity and an extension to compute all optimum solutions.

2.1 Basic Definitions

Definitions and terminology introduced in this work are corrected, and for clarity, modified versions originating from Wu *et al.*, 2014.

A tree $T = (V(T), E(T))$ is a rooted binary tree, where $V(T)$ and $E(T)$ denote the set of

nodes and the set of directed edges (u, v), respectively. By $L(T)$ we denote the set of leaves and by $I(T)$ the set of internal nodes of T, i.e., $V(T) \setminus L(T)$. Let $r(T)$ denote the root node. By $\dot{V}(T)$ we denote the set $V(T) \setminus \{r(T)\}$. For a node v, $c(v)$ is the set of children of v (note that $c(v)$ is empty if v is a leaf), $p(v)$ is the parent of v, and $e(v)$ denotes the branch $(p(v), v)$. Let $T(v)$ be the (maximal) subtree of T rooted at v.

Let $\leq_T$ be the partial order on $V(T)$, such that $u \leq_T v$ if and only if u is on the path between $r(T)$ and v, inclusively.

A *species tree* S represents the relationships among a group of species, in which each leaf is a species. A *gene tree* G is a tree in which each leaf is a gene labeled by a species from which it was sampled. From now on, we assume that for every pair of G and S, the set of all species labels from G is a subset of $L(S)$.

The least common ancestor mapping, $M\colon V(G) \to V(S)$, from gene tree nodes to species tree nodes is defined as follows: if g is a leaf node, then $M(g)$ is the species-label of g (i.e., a leaf from S); if g has two children g' and g'' then $M(g)$ is the least common ancestor of $M(g')$ and $M(g'')$.

2.2 DLC-model

Definition 2.1. (DLC scenario) Given a gene tree G and a species tree S, the *DLC (reconciliation) scenario* for G and S is a tuple $\langle \mathcal{M}, \mathcal{L}, \mathcal{O} \rangle$, such that

- $\mathcal{M}\colon V(G) \to V(S)$ denotes a **species map** that maps each node of gene tree to a species node. In this work, species maps are fixed to the least common ancestor mapping.
- $\mathbb{L}$ denotes the **locus set**.
- $\mathcal{L}\colon V(G) \to \mathbb{L}$ is a surjective locus map that maps each node of gene tree to a locus.
- For a species node s, let parent_loci(s) be the set of loci that yield a new locus in s defined as $\{\mathcal{L}(p(g))\colon g \in \dot{V}(G),\ \mathcal{M}(g) = s \text{ and } \mathcal{L}(g) \neq \mathcal{L}(p(g))\}$. Then, $\mathcal{O}$ is a partial

order on $V(G)$, such that, for every s and every $l \in \text{parent\_loci}(s)$, $\mathcal{O}$ is a total order on the set of nodes $O(s,l) := \{g\colon g \in \dot{V}(G),\ \mathcal{M}(g) = s \text{ and } \mathcal{L}(p(g)) = l\}$.

Subject to the constraints.

1. For every locus l, the subgraph of the gene tree induced by $\mathcal{L}^{-1}(\{l\})$ is a tree. Moreover, every leaf of such a tree that is also a leaf in G must be uniquely labeled by species.

2. For every $g, g' \in V(G)$ if $g \leq_G g'$, then $g \leq_{\mathcal{O}} g'$.

3. A node g is called *bottom* if no child of g maps to $\mathcal{M}(g)$. Then, for every bottom node $x \in O(s,l)$, every non-bottom node $y \in O(s,l)$, we have $x >_{\mathcal{O}} y$. Additionally, if s is not the root of S, then for every bottom node z mapped to $p(s)$, $y >_{\mathcal{O}} z$.

The first constraint assures that all gene nodes with the same locus form a connected component; i.e., each locus is created only once. The second constraint incorporates the gene tree's topology in partial order $\mathcal{O}$. Finally, the third constraint guarantees that all nodes are properly ordered by $\mathcal{O}$.

Inserting Implied Speciation Nodes. For the proper embedding of a gene tree into a species tree, we require additional degree-two nodes inserted into the gene tree.

Given a gene tree, we define the transformation called *insertion of an implied speciation* as follows. The operation subdivides an edge $(g, g') \in G$ with a new node h, called an *implied speciation*, and sets $\mathcal{M}(h) = p(\mathcal{M}(g'))$ if (i) either $p(\mathcal{M}(g')) > \mathcal{M}(g)$, or (ii) $p(\mathcal{M}(g')) = \mathcal{M}(g)$ and g is not a bottom node of $\mathcal{M}(g)$. Note that h becomes a bottom node after the insertion.

Then, we transform G by a maximal sequence of implied speciation insertions. It is not difficult to see that the resulting gene tree with implied speciation nodes is well defined and unique.

Counting evolutionary events. We first define the species map $\mathcal{M}$, then we transform the gene tree by inserting the implied speciation nodes. Next, we define the locus map and

partial order $\mathcal{O}$ on the transformed gene tree. Finally, having the DLC scenario, we can define the evolutionary events induced by the scenario.

We start with several definitions. Let s be a node from the species tree. By $\bot(s)$ we denote the sets of all bottom nodes mapped to s. Similarly, the set of *top* nodes of s is defined as $\top(s) := \bot(p(s))$ if s is not the root, and $\top(r(S)) := \emptyset$, otherwise. By $nodes(s)$ we denote the set of gene nodes mapping to s (i.e., $\mathcal{M}^{-1}(\{s\})$. The *internal* nodes of s are defined as $int(s) := nodes(s) \setminus \bot(s)$.

For G, S and $\alpha = \langle \mathcal{M}, \mathcal{L}, \mathcal{O} \rangle$, we have the following evolutionary events at $s \in V(S)$.

- **Duplication:** A non-root gene tree node g is called a *duplication* (at $\mathcal{M}(g)$) if $\mathcal{L}(g) \neq \mathcal{L}(p(g))$. Additionally, we call g the *locus root*. We then say that a duplication happened on edge $(p(g), g)$.

- **Loss:** A locus l is *lost* at s if l is present in s or at the top of s but l is not present at the bottom of s. Formally, l is lost if $l \in \mathcal{L}(\top(s) \cup nodes(s))$ and $l \notin \mathcal{L}(\bot(s))$.

- **ILS at speciation:** Let $C(s, l)$ be the set of all gene lineages (g, g') such that g is a top node at s, whose loci is l, and g' is mapped to s. Then, locus l induces $\max\{|C(s, l)| - 1, 0\}$ (deep) coalescence events at speciation s.

- **ILS at duplication:** For each duplication d, whose parent has loci l, a gene lineage in species $\mathcal{M}(d)$ at locus l is *contemporaneous* with d if the lineage starts before and ends after the duplication node d. Let $K(d)$ denote the set of all edges contemporaneous with d. Formally, $K(d) = \{g \colon g \in O(\mathcal{M}(d), \mathcal{L}(p(d))$ and $g >_{\mathcal{O}} d >_{\mathcal{O}} p(g)\}$. Then, the duplication d induces $\max\{|K(d)| - 1, 0\}$ (deep) coalescence events.

Problem 1 (DLCParsimony)**.** *Given G, S, and real numbers c_D, c_L, and c_{DC}, the reconciliation cost for a DLC scenario $\alpha = \langle \mathcal{M}, \mathcal{L}, \mathcal{O} \rangle$ is*

$$R_\alpha := \sum_{s \in V(S)} c_D \cdot nD_\alpha(s) + c_L \cdot nL_\alpha(s) + c_{DC} \cdot (nCS_\alpha(s) + nCD_\alpha(s)),$$

where $nD_\alpha(s)$, is the total number of duplication nodes at s, $nL_\alpha(s)$ is the total number of lost loci at s, and $nCS_\alpha(s)$ is the total number of coalescence events at speciation s, and $nCD_\alpha(s)$ is the total number of coalescence events at duplications mapped to s in the scenario α.

2.3 Unconstrained ILP formulation

We now present an Integer Linear Programming (ILP) formulation for solving the DLCParsimony problem. From now on, we refer to this formulation as *M0*.

2.3.1 Model Parameters

First, we define global parameters that can be used to constraint the formulation (see constrained models in the next section).

D_g Binary parameter for each $g \in I(G)$. It is 1, if a duplication event is *allowed* in one of the children of g. In this section $D_g = 1$ for all g, since we do not want to constrain our model.

2.3.2 Model Notation

We now define the notation that will be used throughout the formulation.

$\mathcal{I}(s)$ Possible order values (indices) of gene nodes within a total ordering of gene nodes induced by $\mathcal{O}$ and restricted to species node s. That is, $\mathcal{I}(s) = \{1, ..., |int(s)|\}$

$\mathcal{N}$ The maximum possible number of loci; i.e., maximum possible number of duplications plus one. In particular, $\mathcal{N} = 1 + \sum_{g \in I(G)} D_g$. Further, we denote the set $\{1, \ldots, \mathcal{N}\}$ by $[\mathcal{N}]$.

F_g Indicates the locus index of node g and is defined as $F_g := \sum_{g' \in I(G), g' \leq_{ord} g} D_{g'}$, where $\leq_{ord}$ is some total order on $I(G)$. F_g guarantees that duplication at node g yields a new and distinguished locus F_g in the locus tree.

2.3.3 Decision Variables

Now we declare the core variables needed for the ILP formulation.

x_{uv} A binary variable for edge $(u, v) \in E(G)$. Equals to 1 if v is a duplication; otherwise 0.

y_{gl} Binary variable. 1 if node $g \in V(G)$ is assigned to locus l; otherwise 0.

e_{ls} Binary variable. 1 if locus l is lost at species node/branch s; otherwise 0.

c_{ls} The number of deep coalescence events at a speciation s induced by the locus l.

d_{gl} If g is a duplication and $l = \mathcal{L}(p(g))$, then it denotes the number of corresponding deep coalescence events induced by locus l. Otherwise, $d_{gl} = 0$.

z_{go} Binary variable. 1, if node $g \in V(G)$ is assigned to order $o \in \mathcal{I}(\mathcal{M}(g))$.

w_{gol} Binary variable. 1, if node $g \in V(G)$ is assigned to order o and locus l.

m_{gol} Binary variable. 1, if node g is assigned to order o and locus l and one of children of g is a locus root (i.e., a duplication event happened immediately below g).

2.3.4 Model constraints

Finally, we describe the objective function and the model constraints using the above variables. In particular, the objective function in Equation (1) (below) minimizes the DLC score. The first term in the objective function calculates the total number of duplication events, whereas the second term computes the number of loss events and coalescence events at speciations. The coalescence events at duplications are computed by the last term in the objective function.

$$\begin{aligned}
\min \quad & \zeta = \textstyle\sum_{e\in E(G)} x_e + \sum_{s\in V(S)} \sum_{l\in[\mathcal{N}]} (e_{ls} + c_{ls}) \\
& \quad + \textstyle\sum_{s\in V(S)} \sum_{l\in[\mathcal{N}]} \sum_{g\in int(s)} d_{gl} && (1) \\
\text{s. t.} \quad & \textstyle\sum_{e=(g,g')\in E(G)} x_e \le D_g && g \in V(G) && (2) \\
& \textstyle\sum_{g\in\perp(s)} y_{gl} \le 1 && \forall s \in L(S), l \in [\mathcal{N}] && (3) \\
& \textstyle\sum_{l\in[\mathcal{N}]} y_{gl} = 1 && \forall g \in V(G) && (4) \\
& y_{r(G),1} = 1 && && (5) \\
& F_g x_e \le \textstyle\sum_{l\in[\mathcal{N}]} l y_{g'l} \le F_g x_e + \mathcal{N}(1 - x_e) && \forall e = (g, g') \in E(G) && (6)
\end{aligned}$$

$$-\mathcal{N}\, x_{gg'} \leq y_{g'l} - y_{gl} \leq \mathcal{N}\, x_{gg'} \qquad \forall (g,g') \in E(G), l \in [\mathcal{N}] \tag{7}$$

$$\sum_{g \in \top(s)} y_{gl} - |V(G)|(e_{ls} + \sum_{g \in \perp(s)} y_{gl}) \leq 0 \qquad \forall l \in [\mathcal{N}], s \in V(S) \tag{8}$$

$$\sum_{g \in \top(s),(g,g') \in E(G), g' \in nodes(s)} y_{gl} - 1 \leq c_{ls} \qquad \forall l \in [\mathcal{N}], s \in V(S) \tag{9}$$

$$\sum_{o \in \mathcal{I}(s)} z_{go} = 1 \qquad \forall s \in V(S), g \in int(s) \tag{10}$$

$$\sum_{g \in int(s)} z_{go} = 1 \qquad \forall s \in V(S), o \in \mathcal{I}(s) \tag{11}$$

$$\sum_{o' \in \mathcal{I}(s), o' \leq o} z_{g'o'} \leq 1 - z_{go} \qquad \forall s \in V(S), g, g' \in int(s), (g,g') \in E(G), o \in \mathcal{I}(s) \tag{12}$$

$$2w_{gol} \leq y_{gl} + z_{go} \leq 1 + w_{gol} \qquad \forall s \in V(S), l \in [\mathcal{N}], g \in int(s), o \in \mathcal{I}(s) \tag{13}$$

$$\sum_{g \in \top(s),(g,g') \in E(G), g' \in nodes(s)} y_{g'l} - 1 \leq n_{ls} \qquad \forall l \in [\mathcal{N}], s \in V(S) \tag{14}$$

$$n_{ls} + \sum_{g' \in int(s) \setminus \{g\}} \sum_{o' < o} (w_{g'o'l} - m_{g'o'l}) \leq d_{gl} + |\top(s)|(1 - m_{gol}) \qquad \forall l \in [\mathcal{N}], s \in V(S), o \in \mathcal{I}(s), g \in int(s) \tag{15}$$

$$2m_{gol} \leq w_{gol} + \sum_{e=(g,g') \in E(G)} x_e \leq 1 + m_{gol} \qquad \forall s \in V(S), l \in [\mathcal{N}], g \in int(s), o \in \mathcal{I}(s) \tag{16}$$

$$d_{gl}, e_{ls}, c_{ls}.n_{ls} \geq 0 \tag{17}$$

$$m_{gol}, w_{gol}, x_e, y_{gl}, z_{go} \in \{0,1\} \tag{18}$$

In a most parsimonious reconciliation scenario for each internal gene node g only one of its children can be a new locus root (Wu *et al.*, 2014). This condition is enforced by inequality 2. Inequality 3 enforces that extant gene nodes mapping to the same extant species must be assigned to different loci. Further, each gene node must be assigned to one locus and it is enforced by Constraint 4. Constraint 5 assigns the original locus (locus 1) to the root of the gene tree. Constraint 6 forces the child gene and its parent to map to different loci if there exists a duplication event between them. Constraint 7 guarantees that if there is no duplication event at gene edge (g, g'), then the locus of g and g' must be the same.

Constraint 8 enforces the correct calculation of loss events. In particular, it ensures that e_{ls} for locus l and species s is 1 if there exists a gene node from $\top(s)$ with locus l, while there is no gene node in $\bot(s)$ with the same locus. Constraint 9 ensures the correct assignment of c_{ls} variables (i.e., the number of coalescence events at speciations). Constraints 10 and 11 jointly assign the partial orders to interior nodes at each species branch. Based on these constraints each order must be assigned to one interior node and each interior node must be assigned to one position in the order. Constraint 12 corresponds to constraint 2 in Definition 2.1. Constraint 13 ensures the proper assignment of the w_{gol} variables. Constraints 14 and 15 should be considered together (note that n_{ls} is an additional variable that joins those two equations; it is required to properly account for extra gene lineages at duplications). Those constraints together ensure the proper counting of the deep coalescence events at a duplication that occurs in one of the children of node g for locus l at species node s. Constraint 16 assures the correct assignment of m_{gol} variables.

2.4 Designing efficiently computable models

Here we describe four additional ILP formulations (M1-M4) based on the observations described in the introduction.

2.4.1 Leveraging duplication evidence

While the base model is highly flexible in terms of edges, where duplications can appear, this flexibility contributes substantially to the computational complexity of M0 (see the scalability study for more details). Therefore, in this section, we consider a strategy of restraining the duplication placement only to those edges, where there is *evidence* that a duplication has occurred.

In particular, we call a node $g \in V(G)$ with children g' and g'' an *apparent duplication parent* if $\mathsf{clu}(g') \cap \mathsf{clu}(g'')$ is not empty, where $\mathsf{clu}(v)$ is the set of all leaf labels reachable from v. That is, there exist extant species, which both child lineages of g sort out to.

We then constraint the M0 in the way that only children of apparent duplication parents can be locus roots. In fact, there are two options for how this constraint can be implemented, which we call M1 and M2 that are formalized below.

M1 (restricted locations of duplications). Observe that D_g variables defined in the previous section allow us to constrain the locations of gene duplication events easily. That is, we define the M1 formulation by properly setting the D_g variables: $D_g = 1$ if and only if g is an apparent duplication parent.

M2 (forced locations of duplications). Since apparent duplication parents provide strong evidence of duplications, we define, in addition, a tighter model (M2). In this model, we require that one of the children of each apparent duplication parent *must* be a duplication. Note that, while this is a strong condition, it allows us to simplify the ILP formulation and reduce the number of variables. That is, we anticipate that M2 formulation performs fastest in practice.

More precisely, in this model, we "know", where duplications must appear (at least we know the parents of duplications). Therefore, Inequality (2) in M0 should become an equality (which tightens the solution space); further, the m_{gol} variables become redundant, so they can be removed.

2.4.2 Leveraging the molecular-dating of coalescence events

The space of potential combinations of node orderings greatly influences the computational complexity. Therefore, we propose a strategy to restrict the orderings to a predefined sequence. Such a sequence then can be inferred using the standard molecular-dating methods (Suchard *et al.*, 2018; Sagulenko *et al.*, 2018).

M3 (restricted order). The predefined order is incorporated into the M0 model by the addition of constraints for a variable z_{go}. If node $g \in V(G)$ is assigned to an order $o \in \mathcal{I}(\mathcal{M}(g))$, we add the following constraints $z_{go} = 1$ and $z_{go'} = 0\ \forall o' \neq o$.

M4 (restricted order and locations of duplications). This model is designed to restrict

the order and duplication locations. Analogically as in the M3 model we add the constraints for the predefined order, but in this case, we extend the restricted M1 model.

2.5 Size of ILP formulations

We analyze the size of our ILP formulations in terms of their number of variables and constraints. Let n denote the number of nodes in the gene tree and let m denote the number of nodes in the species tree. Further, let k denote the maximum possible number of loci in the gene tree. Note that $k < n$ and k in the M1 and M2 models can be expected to be significantly smaller than in the M0 model due to the modified D_g variables.

Then in the M0 and M1 models, the upper bound on the number of variables is

$$2km + (2k + 1)(n + n^2) = O(k(m + n^2)),$$

and the number of constraints is

$$(3k + 1)n^2 + (k + 2m + 3)n + 4mk + 1 = O(kn^2 + m(n + k)).$$

Finally, the M2 model has

$$2km + (2k + 1)n + (k + 1)n^2 = O(k(m + n^2))$$

variables, and

$$(k + 1)n^2 + (k^2 + 2m + 3)n + 4mk + 1 = O(kn^2 + m(n + k))$$

constraints. Observe, that the M2 model has fewer variables than the other two models (while asymptotically the same).

The M3 and M4 models are extended variants of M0 and M1 models, respectively. How-

ever, all additional constraints are in fact variable assignments that reduce the complexity of computation.

2.6 Searching for multiple optimal solutions

The proposed formulations can be extended to detect multiple optimal solutions through an iterative algorithm. At each iteration of that algorithm, our models identify one more alternative optimal solution (if such a solution exists). In particular, for a fixed model, at the first iteration, we solve the original model and save the optimal variables x^*, y^*, and z^* as a part of an optimal solution. To identify a different optimal solution with the same objective value, we add a new constraint such that the ILP model does not repeat identifying previously detected optimal solutions. This constraint is defined as

$$\sum_{e \in E(G)} (x_e - 1)x_e^* + \sum_{g \in V(G)} \sum_{l \in [\mathcal{N}]} (y_{gl} - 1)y_{gl}^* + \sum_{g \in V(G)} \sum_{l \in [\mathcal{N}]} (z_{go} - 1)z_{go}^* \leq -1.$$

We repeat this process as long as the optimal DLC score is the same as the previous iterations.

3 Results

Our experimental evaluation consists of two parts. First, we generated simulated data to study the scalability and sensitivity of our models. Next, we verified how developed methods perform on a large-scale empirical dataset.

3.1 Experimental evaluation of simulated data

We present a broad simulation study that (i) compares the computational efficiency and scalability of the developed ILP models with DP and (ii) validates the accuracy of the constrained ILP formulations. Note that we carry out our studies under varied simulation parameters controlling the rate of duplication/loss events as well as the rate of ILS.

3.1.1 Experimental setup

The process for converting an instance of the DLCParsimony problem to an ILP instance was implemented in Python 3. Then ILP instances were solved with the Gurobi optimizer version 9.0 (Gurobi Optimization, LLC, 2020). As for DP (Wu *et al.*, 2014), we used the exact version of the software DLCPar without the heuristic options for a fair comparison. Further, we set the DLCParsimony cost parameters as $c_D = c_L = c_{DC} = 1$. We performed the experiments on our server with 80 cores of 2.20GHz and 512 GB RAM each.

3.1.2 Simulated data

DLCParsimony instances were generated using the standard *SimPhy* simulator (Mallo *et al.*, 2015). SimPhy works by first simulating a birth-death species tree and then applying the 2-step DLCoal process by Rasmussen et al. (Rasmussen and Kellis, 2012) to simulate the multi-locus gene trees. We use the standard simulation parameters derived from the real-world 16 fungi dataset (Molloy and Warnow, 2019; Du *et al.*, 2019b; Rasmussen and Kellis, 2012). In particular, we follow the parameter settings by Molloy and Warnow (Molloy and Warnow, 2019).

To conduct a comprehensive analysis and adequately evaluate the proposed models, we perform our experiments under various realistic levels of gene duplication and loss (GDL) and incomplete lineage sorting (ILS). More precisely, we use three different GDL levels: 1e-10 duplication&loss events per year (low GDL rate), 2e-10 (moderate GDL rate), and 5e-10 (high GDL rate). Further, we use two different ILS levels by controlling the tree-wide effective population size; i.e., we use the effective population sizes of 1e7 and 5e7 (that correspond to low and moderate ILS levels, respectively, according to (Molloy and Warnow, 2019)).

Finally, we simulated DLCParsimony instances with the number of species varying from 5 to 50. That is, overall, we had $3 \times 2 \times 10 = 60$ different parameter settings for DLCParsimony instances. To ensure consistency, for each of the 60 parameter combinations, we generated 10

independent DLCParsimony instances. Then we executed DP, M0, and four constrained ILP models on each of the 600 generated problem instances. Due to a large number of instances and the models' advanced complexity, we constrained each execution time to 10 minutes.

Note that our simulations generated gene trees with branch lengths. From that information M3 and M4 models inferred the order of nodes.

3.1.3 Simulation Results

We performed experiments to test computational feasibility, verify run-time and scalability, and validate the effectiveness of constrained models. The results are shown in Table 1, Table 2, Table 3, Figure 2, and Figure 3. Please refer to the Discussion section for a detailed analysis of the results.

3.2 Model comparison on empirical data

We evaluated the performance of our five ILP formulations on a standard empirical dataset with $5,351$ gene families in fungi (Butler *et al.*, 2009; Wu *et al.*, 2014). This dataset was frequently used as a baseline for orthology-detection methods (Wapinski *et al.*, 2007; Rasmussen and Kellis, 2012; Wu *et al.*, 2013, 2014). Our objective was to perform a detailed comparison across our proposed models and verify our findings from the simulation study on real empirical data.

3.2.1 Empirical study set up

To evaluate our models, we used the unrooted gene trees from Wu *et al.*, 2014 that were estimated for each gene family using PhyML. Note that PhyML trees have edge lengths, which is crucial for M3 and M4 models that fix the order of coalescence events in the DLC-model. In particular, we use the strict molecular clock assumption, which implies that we can directly use the PhyML edge lengths to determine coalescence times (after rooting the gene trees).

To root the gene trees, we first use the standard midpoint rooting. However, duplication-loss events can largely obfuscate credible midpoint rooting (Mykowiecka and Górecki, 2016). Addressing these shortcomings, we propose a novel rooting approach, called DL-rooting, that is sensitive to these events as well as branch-lengths, as we detail below.

That is, we root the gene trees using the following two options:

(i) *Midpoint rooting.* This is the standard approach to rooting in phylogenetics, where a longest path between two leaves is considered, and the root is placed at the middle point of that path.

(ii) *DL-rooting.* An optimal root in an unrooted gene tree can be identified by minimizing the duplication-loss cost between a rooting of the gene tree and the respective species tree (Górecki and Tiuryn, 2007). However, this strategy usually provides multiple edges, where an optimal root can be placed. Such edges always form a subtree in a gene tree (Górecki *et al.*, 2013). Here, we propose to choose a midpoint root strictly within that optimal subtree of the gene tree (i.e., ignoring all other, sub-optimal edges).We implemented this new rooting method in the *URec* software package (Górecki and Tiuryn, 2007) (see Appendix A for more details).

We then executed ILP models M0 through M4 on both midpoint and DL rooted trees, limiting the ILP runtime to 10 minutes per gene tree per model. We followed the same execution setup as described in Section 3.1.1.

3.2.2 Empirical study results

As we specified a 10-minute time-limit per gene tree per model, there were 4 gene families in total (0.07% of all gene families) for which at least one ILP formulation failed to complete on time for either the midpoint or DL rooted gene tree. Therefore, to properly compare our formulations, we excluded these 4 gene families from consideration. Further, there were 15 other gene families, for which the DL-rooting resulted in a multifurcated gene tree (i.e.,

midpoint root coincided with a gene tree node). We had to additionally exclude these 15 additional gene families, since our ILP formulations are currently applicable to fully bifurcating trees only. Table 4 then summarizes the obtained results, and we discuss this table in detail in Section 4.4.

4 Discussion

4.1 Computational feasibility comparison

Table 1 shows each algorithm's breakdown on how many instances it failed to complete within 10 minutes.

We distinguished two measures: TIME for the total time of the test (including pre- and post-processing) and the TIMEILP for the ILP part's computation only (that is, model creation and solving). The execution of the ILPs was terminated when TIMEILP reached 10 minutes. Therefore, we divided the failed tests into three groups. Group A consists of tests where we did not obtain any solution after a forced termination. Group B, is where TIMEILP reached 10 minutes, but we obtained a (potentially non-optimal) solution from the solver. Finally, Group C is where TIMEILP is within the 10-minute limit, while TIME is not. Note that despite the forced termination, we were able to compute optimal solutions for the Group C instances.

As expected, we observed that the constrained ILP formulations generally performed faster than both DP and M0, particularly for instances with more than 50 genes. Overall, M1, M2, and M4 were able to compute the solutions for all instances in time. M0 and M3 were not able to complete within 10 minutes on 19 and 17 instances out of 600, respectively. However, only 5 instances for M0, and 4 instances for M3 are in Group A. Further, from the 10 Group B tests for M0, 3 produced an optimal reconciliation despite the forced termination. In summary, there were only 12 instances out of 600, where M0 did not obtain the optimal reconciliation within 10 minutes. However, in further analysis, we consider all instances that

fall into Groups A, B, and C as 'failed' for a fair comparison with DP.

While DP was faster than M0 on smaller instances, it failed to complete on significantly more tests than M0 (33/600 versus 19/600). We observed that there were 18 instances, where M0 was able to complete, while DP failed. At the same time, there were 4 instances, where DP was able to complete, while M0 failed. However, for all those 4 instances M0 found an optimal solution. In fact, 3 out of 4 of these instances are in Group C for M0, which means that the ILP solver computations finished within 10 minutes. The last instance was in Group B, but despite the forced termination of the ILP solver, M0 produced an optimal reconciliation on that instance.

In summary, the M0 algorithm provided the optimal solution for all instances, where DP succeeded, and was able to solve additional 18 instances, where DP failed.

4.2 Run-time comparison and scalability analysis

Figure 2 demonstrates M0 and M3's scalability using examples of high-GDL and high-ILS levels (we exclude the 10 failed tests for M0, see the last row in Table 1). In general, M0 and M3's runtime was comparable, with the runtime of M3 being slightly faster. The speed improvement enabled the computation of 2 additional instances within 10 minutes by M3 (see Table 1).

Figure 3 presents the runtime results for the algorithms M1, M2, and M4 on the same set of examples. We excluded the runtime of the other three (slower) algorithms from this figure for a better resolution. Figure 3 suggests that the runtime of M1, M2, and M4 was generally comparable, with M2 being the fastest on average and M4 slightly improving over M1.

Finally, the total runtime improvement of the constrained ILPs over the unconstrained ILP (M0) can be seen from Table 2.

4.3 Validating the results of constrained models

Given that for the vast majority of instances M0 or DP have completed, we were able to validate the constrained models' assumptions. That is, we compare the optimum DLC reconciliation score from the constrained models against the overall optimum DLC score (in the unconstrained case). See Table 3 for the results breakdown.

Interestingly, we observed that in all instances (where we know the optimum unconstrained cost) M3 provided exactly the same reconciliation cost as the original DLC-model. M1 and M4 results differed from the 'ground truth' on 10 instances. Crucially, on these 10 instances, M1 and M4 costs were only slightly higher than the optimum reconciliation costs (the difference was *at most* 2). In summary, M1 and M4 provided optimal reconciliation costs in 98.3% of instances.

The fastest ILP, M2, provided over-estimated reconciliation costs more frequently. It was correct in 89.5% cases, and in the other 61 it was only off by at most 8 duplication/loss/coalescence events.

In summary, M3 proved to be highly accurate while providing the opportunity to compute more instances in comparison to unrestricted M0. However, when dealing with larger instances in practice (e.g., more than 200 genes), we recommend using the M1 or M4 formulations that proved to be both very effective and efficient, almost always providing the globally optimum reconciliation cost.

4.4 Model comparison on empirical data

Table 4 compares our five ILP formulations in detail. Primarily, it shows the total DLC cost summed up over all gene families under consideration. A smaller DLC cost implies a more parsimonious reconciliation. To disambiguate the unified DLC costs, we also consider each particular type of events (duplications, losses, deep coalescence at speciations, and deep coalescence at duplications) individually. Further, following Wu *et al.*, 2014, we compute the average duplication consistency score (Vilella *et al.*, 2009) for each model. Given a

duplication node g in a gene tree, the duplication consistency score is a fraction of species that are common between the two child subtrees of g; this score is then averaged over all duplication nodes in a gene tree. Note that if a tree has no duplications, then we say that the duplication consistency score for that tree is 1. Finally, we compare the total runtime of the ILP solver under each model.

First of all, from Table 4 we observe that DL-rooting corresponds to lower DLC costs compared to the midpoint rooting. This suggests that rooting the gene trees using duplication-loss reconciliations is a significantly better choice than the standard midpoint rooting for orthology inference.

Next, as expected, we note that M0 performs better than other models under the overall DLC costs. This is because M0 is the most general model. We then observe that the total DLC cost for M3 is only insignificantly higher than for M0 under both midpoint and DL rootings (M3 DLC cost is less than 0.02% higher than M0's). Moreover, solving M3 was more than twice faster than solving M0. That is, the coalescence order constraints enforced in M3 (using the strict molecular clock assumption) proved to give us a notable practical advantage over M0.

A comparison of the M1, M2, and M4 models to M0 mainly confirms our observations from the simulation study. In particular, we see that M1 performs significantly better than M2, and the difference in the DLC score between M1 and M0 was only 1.3% for midpoint rooting and 1.2% for DL-rooting. Further, we observe no significant difference between M1 and M4, except for runtime. In particular, M4 was more than 10 times faster than M1 and more than 70 times faster than M0 for both rooting-types.

The results for M3 and M4 suggest that constraining coalescence orders is a strongly beneficial strategy for handling large gene families under the DLC-model. Further, combining duplication constraints with coalescence order constraints (i.e, M4) gives the best trade-off in accuracy versus runtime.

Regarding the duplication consistency score, we note that M1 and M4 performed best

here. This is not surprising, since the duplication constraints implemented in M1 and M4 limit the duplication locations to only those gene nodes, where there is an overlap in species between the childrens' subtrees. That is, the duplication constraints align with the idea of Vilella *et al.*, 2009 that considers high species overlap as an indicator of a likely duplication.

Finally, we observe that computing DLC reconciliations for DL-rooted trees was noticeably more computationally intensive than computing reconciliations for midpoint rooted trees for each model.

5 Conclusion

We developed a flexible ILP formulation for the popular DLC-model. In simulations we demonstrated that this ILP generally outperforms the state-of-the-art dynamic programming solution by Wu *et al.*, 2014. However, despite this advancement, the ILP still cannot handle large gene families with more than 200 genes due to the advanced computational hardness of DLC-model. Therefore, we proposed and comprehensively tested four additional constrained models. We showed that constraining duplication locations in the DLC-model renders a significantly more scalable ILP without sacrificing the accuracy in a broad simulation study.

Further, while the idea of fixing an order of the coalescence events in the gene trees did not result in a drastic runtime reduction in the simulation study, we observed that this constraint provided a significant advantage over M0 on real empirical data.

In summary, we propose M1 for reconciliation and ortholog identification in large gene families. Further, when molecular dating of gene trees is possible, M4 will be the most optimal choice. In fact, in the empirical study we showed that for accurate reconciliation with M3 and M4, it is sufficient to use the standard edge lengths provided by the popular maximum-likelihood or Bayesian tree estimation methods. At the same time, for smaller gene families, the unconstrained ILP will always provide the optimal solution.

Finally, our novel DL-informed rooting strategy described in Appendix A and imple-

mented in URec, showed a great promise for orthology detection in practice.

Acknowledgements

We would like to thank Javad Ansarifar and Mukul Bansal for their support in building the foundations of the initial constrained DLC-models and helpful discussions. We also thank Jerzy Tiuryn for his thoughtful and detailed comments on the DLC-model definition. OE is supported by the National Science Foundation Grant No. 1617626. PG and JP are supported by the National Science Center grant 2017/27/B/ST6/02720.

References

Ansarifar, J., Markin, A., Górecki, P., and Eulenstein, O., 2020. Integer linear programming formulation for the unified duplication-loss-coalescence model. In Cai, Z., Mandoiu, I., Narasimhan, G., Skums, P., and Guo, X., eds., *Bioinformatics Research and Applications*, 229–242. Springer International Publishing, Cham.

Arvestad, L., Berglund, A.-C., Lagergren, J., and Sennblad, B., 2004. Gene tree reconstruction and orthology analysis based on an integrated model for duplications and sequence evolution. In *Proceedings of RECOMB'04*, 326–335.

Butler, G., Rasmussen, M. D., Lin, M. F., Santos, M. A., Sakthikumar, S., Munro, C. A., Rheinbay, E., Grabherr, M., Forche, A., Reedy, J. L., *et al.*, 2009. Evolution of pathogenicity and sexual reproduction in eight candida genomes. *Nature* 459, 657–662.

Carothers, M., Gardi, J., Gross, G., Kuze, T., Liu, N., Plunkett, F., Qian, J., and Wu, Y.-C., 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*, 1–12.

Chan, Y., Ranwez, V., and Scornavacca, C., 2017. Inferring incomplete lineage sorting, duplications, transfers and losses with reconciliations. *Journal of Theoretical Biology* 432, 1–13.

Du, H., Ong, Y. S., Knittel, M., Mawhorter, R., Liu, N., Gross, G., Tojo, R., Libeskind-Hadas, R., and Wu, Y.-C., 2019a. Multiple optimal reconciliations under the duplication-loss-coalescence model. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* .

Du, P., Hahn, M. W., and Nakhleh, L., 2019b. Species tree inference under the multispecies coalescent on data with paralogs is accurate. *bioRxiv* 498378.

Du, P. and Nakhleh, L., 2018. Species tree and reconciliation estimation under a duplication-loss-coalescence model. In *Proceedings of the 2018 ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, 376–385.

Du, P., Ogilvie, H. A., and Nakhleh, L., 2019c. Unifying gene duplication, loss, and coalescence on phylogenetic networks. In Cai, Z., Skums, P., and Li, M., eds., *Bioinformatics Research and Applications*, 40–51. Springer, Cham.

Goodman, M., Czelusniak, J., Moore, G., Romero-Herrera, A., and Matsuda, G., 1979. Fitting the gene lineage into its species lineage. A parsimony strategy illustrated by cladograms constructed from globin sequences. *Sys Zool* 28, 132–163.

Górecki, P., Eulenstein, O., and Tiuryn, J., 2013. Unrooted tree reconciliation: a unified approach. *IEEE/ACM Trans Comput Biol Bioinform* 10, 522–36.

Górecki, P. and Tiuryn, J., 2006. DLS-trees: A model of evolutionary scenarios. *Theor. Comput. Sci.* 359, 378–399.

Górecki, P. and Tiuryn, J., 2007. Urec: a system for unrooted reconciliation. *Bioinformatics* 23, 511–512.

Gurobi Optimization, LLC, 2020. Gurobi optimizer reference manual. URL http://www.gurobi.com.

Koonin, E. V., 2005. Orthologs, paralogs, and evolutionary genomics. *Annu. Rev. Genet.* 39, 309–338.

Li, H., Coghlan, A., Ruan, J., Coin, L. J., Heriche, J.-K., Osmotherly, L., Li, R., Liu, T., Zhang, Z., Bolund, L., *et al.*, 2006. Treefam: a curated database of phylogenetic trees of animal gene families. *Nucleic acids research* 34, D572–D580.

Lynch, M. and Conery, J. S., 2000. The evolutionary fate and consequences of duplicate genes. *science* 290, 1151–1155.

Maddison, W. P., 1997. Gene trees in species trees. *Systematic biology* 46, 523–536.

Mallo, D., de Oliveira Martins, L., and Posada, D., 2015. Simphy: phylogenomic simulation of gene, locus, and species trees. *Systematic biology* 65, 334–344.

Molloy, E. K. and Warnow, T., 2019. FastMulRFS: Statistically consistent polynomial time species tree estimation under gene duplication. *BioRxiv* .

Mykowiecka, A. and Górecki, P., 2016. Bootstrapping algorithms for gene duplication and speciation events. In Botón-Fernández, M., Martín-Vide, C., Santander-Jiménez, S., and Vega-Rodríguez, M. A., eds., *Algorithms for Computational Biology*, 106–118. Springer International Publishing, Cham.

Ohno, S., 1970. *Evolution by gene duplication*. Springer-Verlag, Berlin.

Page, R. D., 1994. Maps between trees and cladistic analysis of historical associations among genes, organisms, and areas. *Systematic Biology* 43, 58–77.

Rasmussen, M. D. and Kellis, M., 2012. Unified modeling of gene duplication, loss, and coalescence using a locus tree. *Genome research* 22, 755–765.

Sagulenko, P., Puller, V., and Neher, R. A., 2018. Treetime: Maximum-likelihood phylodynamic analysis. *Virus evolution* 4, vex042.

Suchard, M. A., Lemey, P., Baele, G., Ayres, D. L., Drummond, A. J., and Rambaut, A., 2018. Bayesian phylogenetic and phylodynamic data integration using BEAST 1.10. *Virus Evol* 4, vey016.

Szöllősi, G. J., Tannier, E., Daubin, V., and Boussau, B., 2014. The Inference of Gene Trees with Species Trees. *Systematic Biology* 64, e42–e62.

Vilella, A. J., Severin, J., Ureta-Vidal, A., Heng, L., Durbin, R., and Birney, E., 2009. Ensemblcompara genetrees: Complete, duplication-aware phylogenetic trees in vertebrates. *Genome research* 19, 327–335.

Wapinski, I., Pfeffer, A., Friedman, N., and Regev, A., 2007. Natural history and evolutionary principles of gene duplication in fungi. *Nature* 449, 54–61.

Wu, T. and Zhang, L., 2011. Structural properties of the reconciliation space and their applications in enumerating nearly-optimal reconciliations between a gene tree and a species tree. *BMC Bioinformatics* 12, S7.

Wu, Y., Rasmussen, M. D., Bansal, M. S., and Kellis, M., 2014. Most parsimonious reconciliation in the presence of gene duplication, loss, and deep coalescence using labeled coalescent trees. *Genome research* 24, 475–486.

Wu, Y.-C., Rasmussen, M. D., Bansal, M. S., and Kellis, M., 2013. Treefix: statistically informed gene tree error correction using species trees. *Systematic Biology* 62, 110–120.

Population size	GDL	Number of Instances						
		M0			M3			DP
		Grp A	Grp B	Grp C	Grp A	Grp B	Grp C	Total
1e7	1e-10	0	0	0	0	0	0	0
1e7	2e-10	1	0	0	1	0	0	1
1e7	5e-10	3	1	2	2	2	2	8
5e7	1e-10	0	0	1	0	0	0	2
5e7	2e-10	0	1	0	0	0	1	3
5e7	5e-10	1	8	1	1	8	0	19

Table 1: **Computational feasibility comparison.** Number of Instances with running time above 600 seconds out of 100 tests for each combination. The number of terminated tests is represented as a sum of Group A (no results), Group B (approximated result), and Group C (optimal result) instances. Note, all tests for M1, M2, and M4 models finished in time.

	M0	M1	M2	M3	M4
Total TIMEILP	1892.806	37.183	12.019	1006.059	14.718
Total TIME	5113.063	393.970	359.783	4206.099	367.369

Table 2: **Total run-time comparison.** Time in seconds for all models on the example of high-GDL and high-ILS instances, where 10 tests for which M0 computation time exceeded 10 minutes are excluded.

Population size	GDL	Number of Instances			
		M1	M2	M3	M4
1e7	1e-10	0/100	0/100	0/100	0/100
1e7	2e-10	0/99	1/99	0/99	0/99
1e7	5e-10	2/94	14/94	0/94	2/94
5e7	1e-10	0/99	7/99	0/99	0/99
5e7	2e-10	3/99	17/99	0/99	3/99
5e7	5e-10	5/90	22/90	0/90	5/90

Table 3: **Score comparison.** Number of Instances, where constrained models M1, M2, M3, and M4 score was larger than the M0/DP score.

Rooting	Model	DLC	Dup	Loss	DC at sp	DC at dup	DCS	ILP time
midpoint	M0	**37,158**	4130	**6497**	25966	563	0.845	5830.55
	M1	37,648	**4121**	8341	24104	1082	**0.861**	1099.61
	M2	38,910	4787	10471	**23306**	**346**	0.856	**44.76**
	M3	37,163	4134	6579	25892	556	0.845	2456.43
	M4	37,651	4125	8356	24097	1073	**0.861**	54.28
DL-rooted	M0	**34,942**	4124	**6132**	24129	555	0.866	9853.51
	M1	35,378	**4119**	7816	22435	1008	**0.881**	1584.17
	M2	36,553	4728	9707	**21762**	**356**	0.875	**81.89**
	M3	34,948	4126	6165	24100	555	0.865	3632.87
	M4	35,382	4120	7822	22431	1009	**0.881**	138.77

Table 4: **Model comparison on the fungi dataset**. The DLC column lists the total number of combined duplication/loss/DC events summed over all gene families. Then the 4 subsequent columns disambiguate those counts into the number of duplications, losses, DC at speciations, and DC at duplications events, respectively. DCS column lists the average duplication consistency scores under each model. Finally, ILP time shows the total runtime that ILP solvers required for each model. The best results within each column are shown in bold (for each rooting separately).

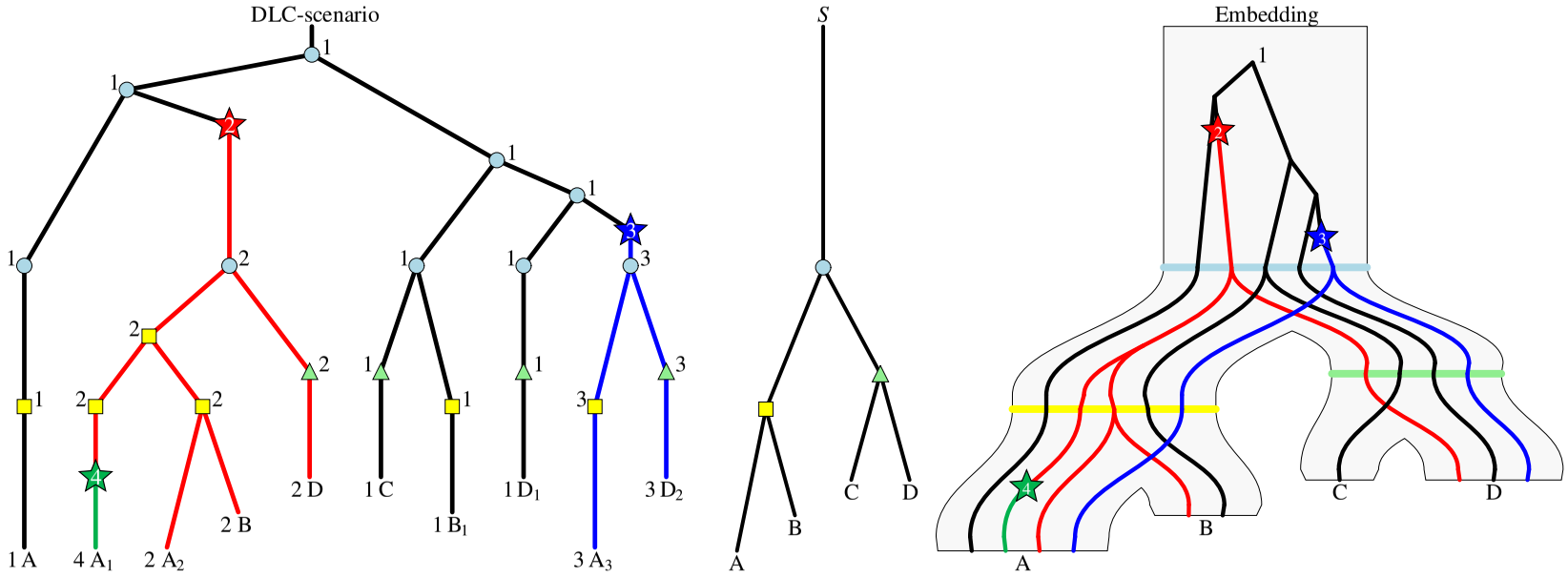

Figure 1: An example of a DLC scenario with four loci 1 through 4. Stars indicate the duplication events. Left: a scenario represented as a gene tree with implied speciations nodes where the nodes are placed with respect to the order $\mathcal{O}$, and the species map to a species tree $S = ((A, B), (C, D))$ (in the middle). Here, a circle in G denotes mapping to the root of S, the square denotes mapping to the root of (A, B), and the triangle denotes mapping to the root of (C, D). The numbers near nodes in G denote locus map. while the stars denote duplication events (2-4). On the right: the embedding of G into S (here, also the nodes are placed with respect to the order $\mathcal{O}$). The scenario has 3 duplications (marked by stars 2-4), 3 lost loci (loci 3 is lost at B and D, loci 2 is lost at C), 3 coalescence events at speciations (1 at A - loci 2, 1 at (A, B) - loci 1, and 1 at (C, D) - loci 1), and 3 coalescence events at duplications (1 for the second duplication and 2 for the third duplication).

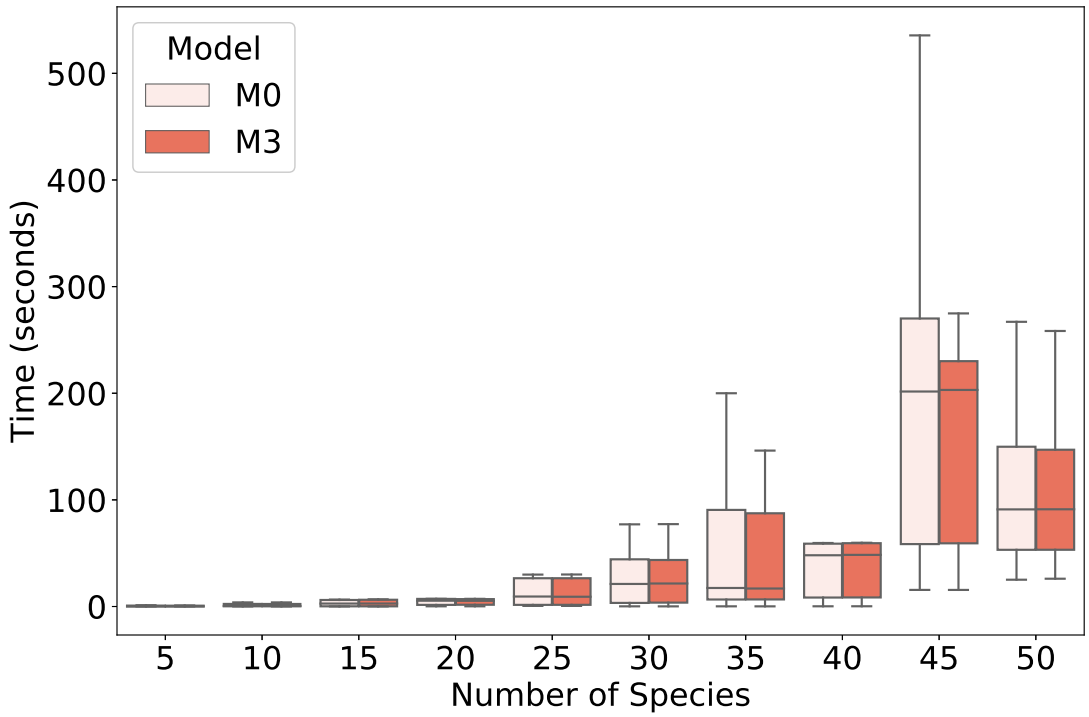

Figure 2: **Computational time comparison for M0 and M3 models.** Results concern 90 not terminated high-GDL and high-ILS instances.

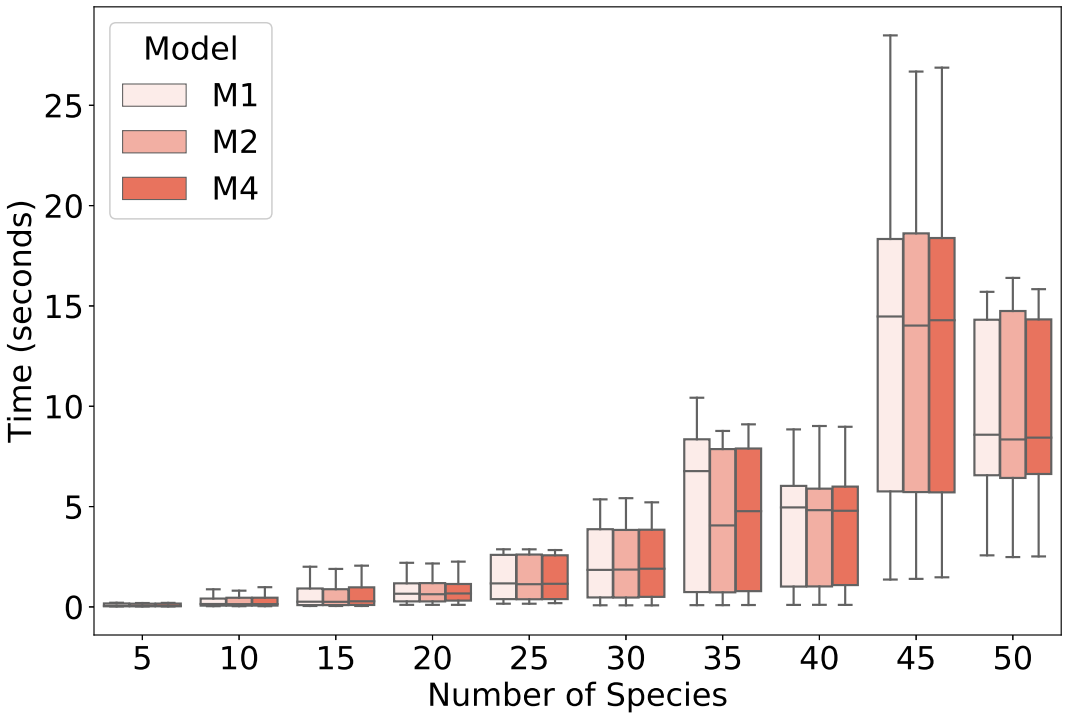

Figure 3: **Computational time comparison for M1, M2, and M4.** Results concern 90 not terminated high-GDL and high-ILS instances.

A DL-rooting with URec

Gene tree parsimony costs, such as the duplication-loss (DL) cost (Page, 1994; Goodman *et al.*, 1979), were defined only for comparing rooted gene trees with rooted species trees. However, in general, unrooted trees can be compared with rooted trees by identifying the rootings of the unrooted tree that is minimizing any provided cost function between a pair of rooted trees. Further, the gene tree parsimony costs satisfy the so-called *plateau property*, which is sufficient for the linear time identification of all optimal rootings and rooting costs in the unrooted gene tree. The plateau property is satisfied when all optimal rootings of the unrooted gene tree form a subtree in this tree, and the rootings along every path toward a leaf have monotonically increased costs (Górecki *et al.*, 2013). Here, we describe a novel DL-rooting strategy for weighted trees, in which the root is defined as the midpoint rooting

of the subtree induced by the plateau edges. This approach is similar to the classic midpoint plateau rooting strategy from (Mykowiecka and Górecki, 2016). However, in difference to this strategy, we consider branch lengths instead of distances measured using edge counts, allowing for a much more informed and refined root determination.

In what follows we detail our new plateau rooting strategy and demonstrate its applicability.

Midpoint DL-plateau rooting

Given a (rooted) gene tree G and a species tree S, by $DL(G, S)$ we denote the duplication-loss cost (Page, 1994) of reconciling G and S. We now summarize how the plateau is identified in an unrooted gene tree, whose definition is analogous the definition of a gene tree presented in Section 2.1, with the difference that the tree has no root. Let U be an unrooted gene tree. If e is an edge of U, by U_e, we denote the rooting of U obtained from U by placing the root on e. Then, given a species tree S, the DL-plateau of U is defined as

$$\text{argmin}_{e \in U} DL(U_e, S).$$

In other words, the DL-plateau determines the best rooting edges by minimizing the duplication-loss cost. However, in general, the plateau may contain more than one edge, thus such rootings may be non-unique. It follows from the theory of unrooted reconciliation that the subtree induced by the DL-plateau form a full subtree of U (Górecki *et al.*, 2013). Thus, if U has branch lenghts, then the DL-plateau midpoint rooting is defined as the midpoint rooting of the subtree induced by the DL-plateau. Note that, similarly to the classical midpoint rooting, the DL-plateau midpoint rooting may be a non-binary tree (e.g., $U = (a : 1, a : 1, a : 1)$), however, it is highly unlikely in the case of empirical datasets.