# Phylogenetic Reconciliation with Transfers

**Han Lai**
hanlai@andrew.cmu.edu

August 4, 2017

Department of Biological Sciences
Carnegie Mellon University
Pittsburgh, PA 15213

**Thesis Committee:**
Dannie Durand, Chair
Luisa Hiller
Russell Schwartz
Lars Arvestad (University of Stockholm)

*Submitted in partial fulfillment of the requirements*
*for the degree of Doctor of Philosophy.*

**Keywords:** phylogenetics, reconciliation, heterogeneity

## Abstract

Correctly inferring the events in the history of a gene family is crucial to relating gene evolution to ecological adaptation, understanding the evolution of gene function, and inferring homology relationships. Information about the evolution of a gene family can be obtained from the incongruence between a gene tree and the associated species tree. Phylogenetic reconciliation algorithms infer gene events through a formal comparison of the gene and species trees.

In this thesis, I expand the reconciliation framework to capture more complex evolutionary scenarios. Current reconciliation algorithms are capable of inferring duplication, transfer, and loss events when both the gene tree and the species tree are well resolved. They are not, however, well equipped to infer events with non-binary trees. A non-binary species tree indicates that evolutionary forces are also in play on the population level. A non-binary gene tree reflects uncertainty in the gene tree branching order, due to insufficient information from the sequence data. Reconciliation algorithms that do not account for these processes can result in incorrect inference of events. To address these problems, I first introduce an algorithm that reconciles a binary gene tree with a non-binary species tree while accounting for gene tree incongruence that could result from population processes rather than gene events. Second, I present an exact algorithm and several fast heuristics that use reconciliation to resolve uncertainty in a non-binary gene tree.

In a parsimony optimization framework, reconciliation seeks the solution that minimizes the weighted sum of the inferred events. One major challenge of this approach is how to select event weights that will infer the correct event history. First, I tackle this challenge from a probabilistic perspective by considering how the underlying gene event rates influence the best choice of event weights. Second, I use a topological approach to identify common tradeoffs between histories with transfers and histories with duplications and losses. By making these tradeoffs explicit, my approach provides a framework for applying biological intuition to the problem of weight selection. This significantly improves the researcher's understanding of how different weights will affect the reconciliation result, leading to better estimation of gene events.

I have implemented the algorithms described above in Notung, a publically available and widely used reconciliation software package. Using this software, I demonstrate the applicability of my theoretical work to concrete biological problems via a phylogenomic analysis in Cyanobacteria. The results reveal a link between gain and loss of photosynthesis genes and niche adaptation.

# Acknowledgments

I would first like to thank my advisor, Dannie Durand, for her support and encouragement. Dannie is not only a great research mentor who teaches me critical thinking and the philosophy of academic research, but also a friendly boss who keeps the lab atmosphere alive and interesting. In my years at Carnegie Mellon University, Dannies has taunt me not only the techniques of doing research, but also the ways of approaching problems. As an international student, writing was one of my weakness and Dannie has helped me with countless papers, presentations, posters, and thesis. Her mentor style is also greatness suitable for me as she fully supported me pursuing a project even though it was not quite related to my main project. In addition to the research skills, she greatly helped me achieve high level of problem solving, critical thinking, open-mindness, and communication skills.

I would also like to thank all my committee members, Luisa Hiller, Russell Schwartz, and Lars Arvestad. Their support and feedback on my yearly committee meeting kept me on track and is greatly helpful for me to overcome several of my greatest obstacles in my research. Without their comments and continuous support, this thesis would not have matured as nicely.

This work was also greatly benefited from so many help people. I would like to thank all the members of the Durand Lab: Maureen Stolzer, Philip Davison, Annette McLeod, Minli Xu, Jacob Joseph, Rosie Alderson, for creating a enjoyable and encouraging lab environment. I would also like to thank my friends who made my years in Pittsburgh a unique and happy experience, and Danyang for helping me with the thesis defence presentation and provided valuable feedback as an fresh eye. Last but not least, I would like to thank my parents, Rong Huang and Xuebo Lai. Without the education and support they provided, and the my personality that they helped constructed, I would not be able to achieve this degree in Carnegie Mellon University.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Fast, cheap sequencing has made it possible to study large collections of gene trees sampled from the same species. The interplay of species and gene evolution creates complex phenomena. The evolution of a group of species can be represented as a species tree, but the trees for each gene family in these genomes may not be the same as the species tree, and may not be the same as each other (Goodman et al., 1979; Maddison, 1997). The phylogenetic data sets from the sample genes and species reveals gene tree heterogeneity. Many forces can contribute to gene tree incongruence and more than one may be acting on any particular data set. There are three main sources of gene tree incongruence: Gene events (duplication, transfer, and loss); population processes (migration, hybridization, introgression, incomplete lineage sorting); and phylogenetic error (incorrect data or insufficient data).

Understanding these incongruences is crucial for reconstructing the history of species and genes within them. My thesis focuses on the problem of resolving the difference between gene tree and species tree topologies to extract evolutionary information. In this chapter, I briefly describe 1) the causes of incongruence between gene trees and species trees: gene events, population processes, and phylogenetic error and 2) the road map of my thesis that describes reconstructing gene history, the open questions and my contributions.

## 1.1. Source of Gene Tree Incongruence

When a group of organisms become isolated, either geographically or genetically, it follows that the genes within those organisms will begin to evolve independently. The process of speciation occurs as these isolated subgroups of the original group then form new species that are different from each other. There are several models of speciation. In the most commonly accepted model for multicellular organisms, speciation occurs when the new species is isolated and changes independently, long enough that they lost the ability to breed with the original species. However, how prokaryotic speciation occurs has been actively discussed and remains an open question (Konstantinidis et al., 2006). If new species evolve independently from each other, it follows that after speciation, the relationship of genes in these new species should also reflect the relationship of the species. The relationships between species that are evolving by vertical descent can be represented using trees. Because speciation events create new species, which abstractly diverge from the lineage of the original species, this phenomenon is consistent with the structure of a tree, in which every node gives rise to two nodes. However, in some cases, even after speciation events, the two new species can still produce new species through hybridization or migration. In this case, a species network is more suitable to represent the relationships (Huson and Scornavacca, 2011).

A gene residing in a genome of a species evolves with the species and experiences the evolutionary steps that the species experiences. That is, if a speciation event occurred to create two new species, the genes in the genome also experience this speciation and evolve independently in the descendants. In addition to speciation, many other processes can act on genes and "create" new genes, including gene duplication and horizontal gene transfer (Goodman et al., 1979; Maddison, 1997). Then from these present species, a set of several similar genes, formed by speciation, duplication, and transfer, can be found as a gene family. Similar to a species tree, the relationships between genes in a gene family can also be represented as trees, in which each split represents divergence of genes that may be caused by co-speciation with species tree, gene duplication, or horizontal gene transfer. This causes the gene tree to disagree with the species tree. Other sources of gene tree incongruence includes population processes and phylogenetic error.

### 1.1.1. Gene Events: Duplication, Transfer, and Loss

Gene duplication has been one of the major forces that drives the evolution of new genes and new functionality in organisms (Magadum et al., 2013; Mendivil Ramos and Ferrier, 2012; Ohno, 1970).

Gene duplication can occur through several molecular processes:

- Retrotransposition is the processes of an mRNA fragment being reverse transcribed into DNA and inserted back into the genome. This process creates a new copy of the original DNA, usually without introns, because intron splicing may have occurred before the retrotransposition.

- Unequal crossing-over occurs when two chromosomes are not lined up correctly during recombination. This will result in a chromosome that contains more genetic material than the original chromosome. The extra material may contain a fragment, a piece of a gene, or one or more whole genes. Thus, one or more duplicated genes in the same chromosomal region are observed.

- Whole genome duplication can occur during cell division, resulting in an organism that contains two copies of its genome from the parent. After the whole genome duplication, the organism's genome is twice the genome size, and it will mostly likely go through a massive gene loss (Inoue et al., 2015; Zheng et al., 2008).

These mechanisms, although all considered as gene duplication, may leave evidence in genome patterns. Retrotransposition produces gene copies without introns in the genome that can be identified by alignment. Unequal crossing over usually leaves the extra gene copy near the original gene, resulting in tandem repeats (Durand and Hoberman, 2006).

Horizontal gene transfer (HGT) is another force that leads to the acquisition of novel genetic material. Gene transfer occurs between different species and strains in the same species. HGT is commonly found in prokaryotes, in which HGT provides a way of introducing and deleting DNA material from the chromosome, effectively changing the ecological and pathological features of the bacteria (Andam and Gogarten, 2011; Lawrence, 1999; Ochman et al., 2000). Recently, accumulating evidence suggests that gene transfer can also occur in fungi (Fitzpatrick, 2012), between bacteria and animals (Dunning Hotopp, 2011) and also in eukaryotes (Andersson, 2009; Gao et al., 2014; Jain et al., 1999; Richardson and Palmer, 2007).

Like gene duplication, HGT in prokaryotes also appears as a result of several underlying molecular mechanisms.

- Transformation is the ability of some bacteria to take a DNA fragment from the environment and incorporate it into their own genome.
- Transduction is a molecular process in a cell, in which foreign DNA material is introduced and inserted into the genome by a virus or transposon.
- Conjugation occurs when a bacterial cell is in direct contact with another bacterial cell. In this case, the genetic material is directly transferred though the cell wall.

Gene duplication and gene transfer are both forces that increase the gene family size, whereas gene loss reduces the gene family size. Gene duplication, transfer, and loss events occurring in bacteria lineages are consistent with the observation that bacterial genomes are in continuous flux(Koskiniemi et al., 2012). Earlier beliefs suggest that the evolution is mostly driven by gene duplication, gene transfer and some other population processes. Recent studies have revealed that gene loss is also acting as a source of genetic variation and causes species diversity (Albalat and Cañestro, 2016).

### 1.1.2. Population Processes

A gene tree, in the absence of gene events, is the history of a locus of individuals as it is passed from parent to child. This history is not guaranteed to be the same as the population history. The history of a locus within a population can be modeled by coalescent theory, which is used to predict the probbability that gene tree disagrees with the species tree under various assumptions. The probability that the branching pattern of a gene and species will agree depends on properties such as population size, selection force, linkage disequilibrium, population structure, and the time between divergences. For example, assuming constant effective population size, no selection, and random mating, the probability that the gene branching pattern disagrees with the species branching pattern depends on the time between divergence. If the time between divergences is long enough, genetic variation present in the population at one divergence will not persist through subsequent divergences. In this case, the gene branching pattern represents the history of the population (Figure 1.1a). In contrast, incomplete lineage sorting (ILS) represents the scenario where the time between speciation events is short, such that the polymorphism may persist through the next specia-

tion event (Figure 1.1b, 1.1c). One of the several models for ILS is proposed by Pamilo and Nei (1988). In a species tree with three leaves, under the assumption of constant effective population size, neutral evolution, no population structure, and random joining of lineages working backward in time, the probability of observing a gene tree that agrees with the species tree is

$$f_C = 1 - \frac{2e^{-T}}{3}. \tag{1.1}$$

Here $T = \Delta t/N_e$ is the branch length in coalescent units, defined as the number of generations between speciation events ($\Delta t$) normalized by the effective population size ($N_e$). The probability of observing either one of the incongruent topologies is

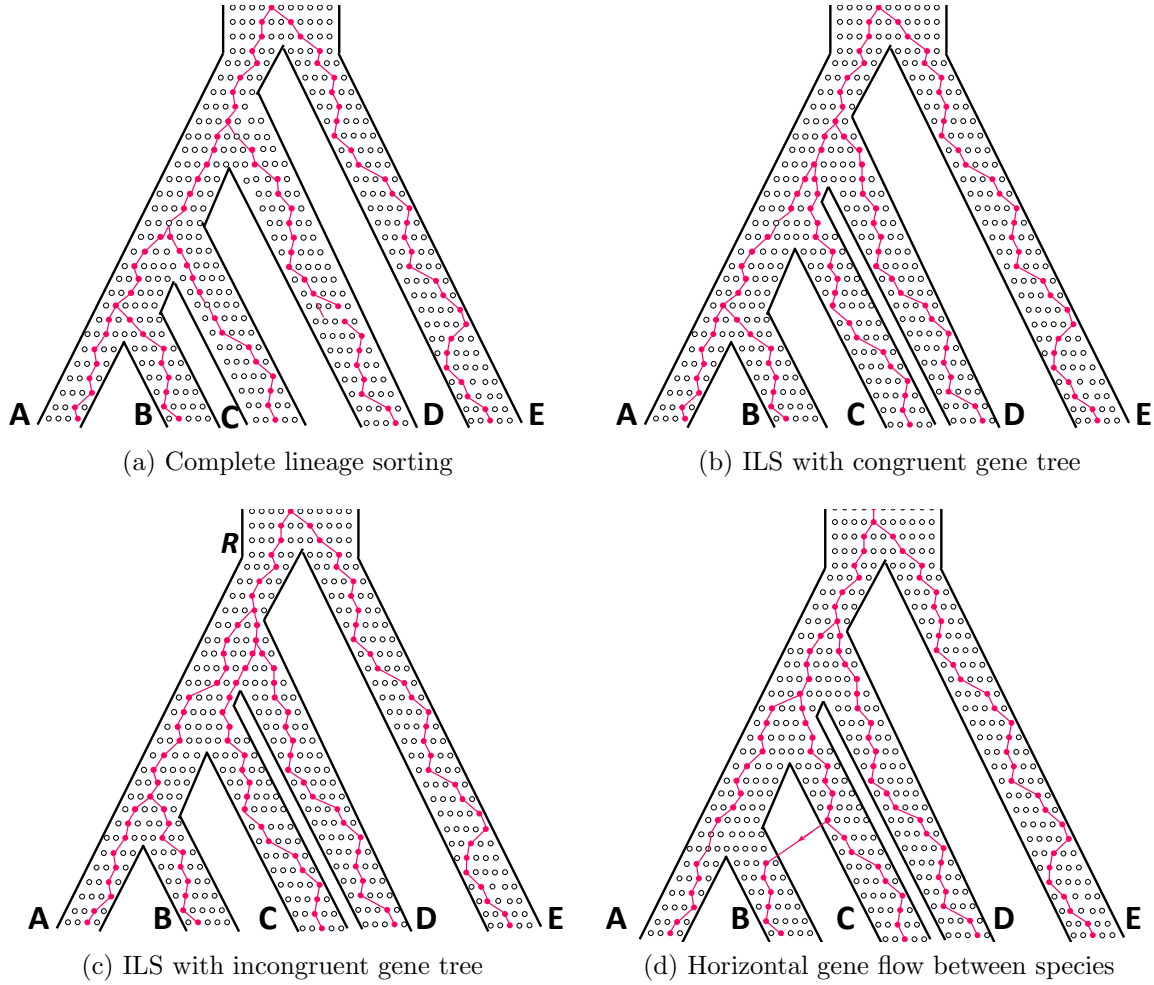
$$f_I = \frac{e^{-T}}{3}. \tag{1.2}$$

When the species tree is a polytomy ($T = 0$), all three topologies are equally probable. When $T > 0$, gene trees with the true topology will occur with the highest frequency and the remaining two topologies will occur with equal frequency (Pamilo and Nei, 1988).

Horizontal population processes, such as migration and hybridization, can also result in loci that have a history different from the population history. They describe the mixture of genetic material between two distinct groups of populations, which introduces alleles with a history that does not reflect the history of the population (Figure 1.1d). The scenario that the genetic variation of previous speciation still exists when the next speciation occurs leads to the situation that the sampled genes from present day populations represent allelic relationships, instead of reflecting the species relationships (Avise et al., 1983, 1984; Maddison, 1997; Nei, 1987; Tajima, 1983; Takahata and Nei, 1985), resulting in incongruence from the species tree (Figure 1.1c, 1.1d). These scenarios, unlike gene duplication, transfer, and loss, are not molecular mechanisms that changes the locus of a gene, but population phenomena that causes gene tree incongruence. If this source of incongruence is not considered, incorrect inferences may occur.

### 1.1.3. Phylogenetic Error

The history of the gene family may be the same as the history of the species, but the gene tree may still disagree with the species tree due to phylogenetic error. In this case, the gene tree




(a) Complete lineage sorting

(b) ILS with congruent gene tree

(c) ILS with incongruent gene tree

(d) Horizontal gene flow between species

Figure 1.1: Example of population processes that result in gene tree to represent the history of the individuals that may or may not be the same as the history of species. The dots in each species tree branch represent individuals of the population. The pink lines indicate the lineages of the gene family.

built from phylogenetic software does not reflect the true history of the gene family. Several common errors that cause the reconstructed gene tree to disagree with the species tree are: sparse taxon sampling, long branch attraction, saturated sites, variation in substitution rates between lineages or over time, and sequence composition bias. In phylogenetic analysis, phylogenetic error should be distinguished from gene events because the incongruence caused by error may lead to incorrect results.

## 1.2. Thesis Outline

Reconciliation is the process of comparing a species tree and a gene tree to infer the events that explain the incongruence. To infer duplication ($D$), transfer ($T$), and loss ($L$) events, reconciliation can be applied to gene tree and species tree under various event models (DL, DTL, TL, etc), and various optimization criteria (maximum likelihood and maximum parsimony). A detail background is provided in Chapter 2. Several of the innovative algorithms that I developed to extend the reconciliation framework to solve more complex problems are included in Chapter 3. These algorithms include DTL reconciliation that incorporates ILS, faster algorithms to validate DTL solutions, and domain tree level reconciliation. Chapter 4 describes a species-tree aware method to fix unresolved gene trees based on the DTL event model.

The reconciliation algorithms under the DTL event model require a set of costs for each gene event. These event costs, provided by the user, are usually hard to choose and crucial to correct inference of the event history. In Chapter 5 and Chapter 6, I approached the problem of selecting cost from two perspectives. I first compared maximum parsimony and maximum likelihood reconciliation and concluded that under several restricted assumptions, the cost can be directly calculated from event rates and the result of MP reconciliation is equivalent to ML reconciliation. I then approach this problem from the perspective of different gene tree topology scenarios that suggest trade-offs between different gene events. This approach allows researchers to apply biological insights to the process of cost selection.

Chapter 7 describes the work I have done in a reconciliation software hosted by the Durand Lab since 2000. I implemented all the algorithms discussed in Chapters 3 and 4. In this chapter, I discuss new functionality tat I added to the software to facilitate phylogenomic reconciliation where many gene trees are reconciled simultaneously. I applied Notung to several data sets and discussed the observations from these results. Lastly, in Chapter 8, I provide an overview of the entire thesis.

## 1.3. Notations

Given a tree, $T_i = (V_i, E_i)$, where $i \in \{G, S\}$ represents a gene tree or species tree, $\rho(T_i)$ denotes the root of $T_i$, and $g \in V_G, s \in V_S$ represent individual gene/species nodes. I use

$L(T_i)$ to denote the leaf node set of $T_i$. The size of the leaf set is denoted $L_N(T_i)$ instead of $|L(T_i)|$ because the vertical bar can be ambiguous when this value is used in a conditional probability. $T_i(v)$ is the subtree of $T_i$ rooted at $v \in V_i$. $l(v), r(v), p(v)$ and $C(v)$ denotes the left child, the right child, the parent, and the set of all children of node $v$. $E_i$ represents all edges in a tree and each edge is denoted $(u, v)$ where $u$ is the parent. For any node $u$, the branch length of $(p(u), u)$ is denoted $t(u)$ instead of $|u|$, again, to avoid the ambiguity with a conditional probability. Given two nodes $u, v \in V_i$, if $u$ is on the path from $v$ to root of $T_i$, then $u$ is an ancestor of $v$, designated $u \geq_i v$, and $v$ is a descendant of $u$, designated $v \leq_i u$. If $v \not\geq_i u$ and $u \not\geq_i v$, $u$ and $v$ are incomparable, designated $u \not\lesseqgtr_i v$. We use $LCA(u, v)$ to represent the least common ancestor node of $u$ and $v$.

Reconciling a gene tree $T_G$ and a species tree $T_S$ results in a set of reconciliation solutions $\gamma$, in which, each solution $\gamma$ describes a mapping $\mathcal{M} : g \to s, g \in V_G, s \in V_S$. This mapping is a many-to-one mapping between gene nodes and species nodes. The reverse mapping is denoted $\mathcal{M}^{-1} : s \to G, G \subset V_G, s \in V_S$.

Table 1.1: Notation

| Symbol | Definition |
|---|---|
| $T_G = (V_G, E_G)$ | Gene tree with nodes $V_G$ and edges $E_G$ |
| $T_S = (V_S, E_S)$ | Species tree with nodes $V_S$ and edges $E_S$ |
| $\|V_i\|$ | Size of tree $T_i$ |
| $L_N(T_i)$ | the number of leaves in tree $T_i$ |
| $t_v$ | the length in time units of the tree branch $(p(v), v)$. |
| $L(T_i)$ | Set of leaf nodes of tree $T_i$ |
| $\rho(T_i)$ | Root of tree $T_i$ |
| $T_i(v)$<br>$T_G(g)$<br>$T_S(s)$ | Subtree of tree $T_i$ rooted at $v$ |
| $g \in V_G$ | Node in the gene tree |
| $s \in V_S$ | Node in the species tree |
| $p(v)$ | The parent of node $v$ |
| $C(v)$ | The set of children nodes of node $v$ |
| $\|C(v)\|$ | The size (number of children) of node $v$ |
| $\{l(v), r(v)\} = C(v)$ | Left and right children of binary node $v$ |
| $< u, v > \in E_i$ | Edge $< u, v >$, where $u$ is the parent of $v$ |
| $u \geq_i v$ | Node $u$ ($v$) is an ancestor (descendant) of node $v$ ($u$) in tree $T_i$ |
| $u \not\lessgtr_i v$ | Nodes $u$ and $v$ in tree $T_i$ are incomparable<br>$u$ is neither an ancestor or descendant $v$ |
| $\mathfrak{T}(T_G)$ | All binary resolutions of the gene tree $T_G$ |
| $T_G^b \in \mathfrak{T}(T_G)$ | A given binary resolution of gene tree $T_G$ |
| $T_G^* \in \mathfrak{T}(T_G)$ | The best binary resolution of gene tree $T_G$ |
| $k(g)$ | Outdegree of node $g$. |
| $k_m$ | Maximal outdegree in the tree. |
| $\mathfrak{T}(g)$ | All binary resolutions of subtree rooted at $g$ |
| $T_g^b \in \mathfrak{T}(g)$ | A given binary resolution of node $k$ |
| $T_g^* \in \mathfrak{T}(g)$ | The best binary resolution of node $k$ |

Table 1.2: Reconciliation Notation

| Symbol | Definition |
|---|---|
| $w = \text{LCA}(u, v)$ | Node $w$ is the common ancestor of nodes $u$ and $v$ |
| $\gamma = (T_G; \mathcal{M}; \mathcal{E}; \Lambda)$ | Reconciliation of $T_G$ |
| $\mathcal{E} : V_G \setminus L(T_G) \to \{S, D, T\}$ | Function indicating event at $V_G$ |
| $\epsilon = \mathcal{E}[g]$ | The divergence at gene $g$ is result of event $\epsilon$ |
| $\mathcal{M} : V_G \to V_S$ | Mapping function of $V_G$ to $V_S$ |
| $s = \mathcal{M}[g]$ | Gene $g$ is mapped to (i.e., occurred in) species $s$ |
| $\mathcal{M}_L : L(T_G) \to L(T_S)$ | Mapping function at leaves |
| $\Lambda \subset E_G$ | The set of transfer edges |
| $\epsilon \in \{S, D, T, L\}$ | event type |
| S, D, T, L | speciation, duplication, transfer, and loss event |
| $C_\epsilon$ | cost of event type $\epsilon$ ($C_D$, $C_T$, or $C_L$) |
| $N_\epsilon$ | number of event type $\epsilon$ ($N_D$, $N_T$, or $N_L$) |
| $\kappa(\gamma)$ | cost of reconciling gene tree $T_G$ with species tree $T_S$ |
| $\mathcal{R}(T_G, T_S)$ | All possible reconciliations |
| $\gamma^*$ | A given reconciliation with minimal cost |
| $\mathcal{R}^*(T_G, T_S)$ | All reconciliations with minimal cost |
| $\mathcal{K}$, $\mathcal{H}$ | cost table and bookkeeping table |
| $\mathcal{K}_g$ | cost array for node $g$ |
| $\mathcal{K}_g[s]$ | cost table entry for node $g$ mapped to node $s$ |
| $\kappa_E$ | The reconciliation cost obtained with the exact method. |
| $\Delta$ | The error, i.e., the normalized difference between the costs obtained with a heuristic and the exact method. |

# Chapter 2

# Background

The ability to accurately infer gene events and population processes can grant new insights into understanding how gene family expansion and gene function evolution are correlated with species evolution, development of metabolic processes, and the ecological relationship betweens species. One common approach to discover the evolutionary processes in a gene family takes advantage of the incongruence between a gene tree and species tree. This approach to reconstruct gene events and/or population processes is called reconciliation. In this chapter, I discuss the history of inferring gene events using the reconciliation approach, including the general problem statement, challenges, and basic algorithm developed by previous work. This provide background of the field for the following chapters, in which I describe how my work fits into the field as a forward step for the open problems.

## 2.1. Reconciliation: Problem Statement

Given a rooted gene tree, a rooted species tree, an event model, and a table indicating the species where each leaf in the gene tree is sampled from, the goal of reconciliation is to find the event history that best explains the topological difference between the gene tree and the species tree, based on a specified optimization criterion (Doyon and Chauve, 2011; Nakhleh, 2010, 2013; Nakhleh and Ruths, 2009).

This general problem statement represents a family of problems that differ in the types of

data, event model, and optimization criteria. For example, the input data, including the gene tree and the species tree, can be in different formats: input trees could be binary or non-binary, and species tree could be undated or dated with relative speciation times. The event model could encompass any combination of duplication (D), transfer (T), and loss (L) events. The optimization criterion may also be either maximum likelihood or maximum parsimony. Each of these pieces are interchangeable and provide applications to many potential questions. Here, I review details of the different options that are covered in my thesis. For brevity, other possible options are not covered; see Doyon and Chauve (2011) for a review.

## 2.1.1. Data Types

### 2.1.1.1. Binary and Non-binary Trees

The simplest form of the input gene tree (species tree) is a binary tree, where each node is a parent of two children nodes and each node represents a gene (species). Non-binary nodes (also called polytomies) in the tree are used to present different information in gene tree and species tree. Two types of polytomies exist. A soft polytomy is the result of insufficient evidence for the branching order. Alternatively, if the true history is a simultaneous separation of three or more children, this results in a hard polytomy.

If a branch in a tree is not well supported by the data, this branch can be collapsed to form a soft polytomy in the tree. In this case, the polytomy represents a set of uncertain binary resolutions, and it is usually required to "fix" the polytomy and find a binary resolution to replace this polytomy before the reconciliation analysis.

In the species trees, hard polytomies may occur due to simultaneous speciation of multiple species. In this case, the hard polytomy is not a representation of unsupported branches but the actual history of the species. These polytomies, in an abstract way, are equivalent to immediate consecutive speciations that result in species branches with 0 length. Incomplete lineage sorting, as described in Chapter 1 may also occur in these species trees.

#### 2.1.1.2. Parasite Tree and Host Tree

The general form of the gene tree-species tree reconciliation problem is very similar to the host-parasite reconciliation problem, and the algorithms developed for reconciliation are also applicable in the host-parasite situation. The phylogeny constructed from parasite species and the phylogeny constructed from host species can be cast to a gene tree and species tree as the input for the regular reconciliation problems, and the result of reconciliation software can be converted to host-parasite events accordingly. In this context, a parasite tree (analogous to a gene tree) is reconciled with a host tree (analogous to a species tree). The reconciliation aims to find host-switching events (analogous to gene transfers), parasite speciation events independent from the host (analogous to gene duplication), and parasite extinction events (analogous to gene loss). Because these two problems are so similar to each other, the algorithms and theoretical results for one problem can be also applied to the other problem. Different event models regarding the host-parasite problem are also described later in Section 2.1.3.5

### 2.1.2. Optimization Criteria

There are two widely used optimization criteria: maximum likelihood (ML) and maximum parsimony (MP). In my thesis, I focus on solving several challenges involving MP optimization criteria in reconciliation. The optimization criterion in each chapter can always be assumed to be MP unless otherwise stated. Under MP optimization, the event history that requires the fewest events to explain the incongruence is the best solution. As this puts the same weight on each event type, a more popular and general MP criterion is to find the solution that minimizes the weighted sum of events:

$$C(N_D, N_T, N_L) = N_D C_D + N_T C_T + N_L C_L, \tag{2.1}$$

where $C_\epsilon$ represents the weight for event type $\epsilon$, and $N_\epsilon$ represents the number of events $\epsilon$ in a solution.

MP and ML reconciliation are based on very different principles and each has its advantages. ML approaches have associated probability support for the reconciliation result, but they generally require more information, such as accurate estimation of branch length as units of time, and a realistic model to estimate the probability of gene subtrees. If the event rate

is unknown, an extra step to use machine learning to estimate event rate is required. This step would require a large number of gene trees to obtain correct estimation of the rate of each type of event. Currently, there is no closed form solution for ML optimization criteria. A general heuristic to find a reconciliation under ML optimization criterion is based on an iterative approach, wherein each iteration, a new reconciliation solution is generated and stored if the likelihood is determined to be greater. The MP model, compared to ML, does not require a probability model for gene subtrees or the rate of events. However, it does require a user-specified cost vector. It also does not require species tree to be dated. However, MP reconciliation can still benefit from speciation times for checking if a solution is infeasible. MP reconciliation also has much better runtime performance than ML algorithms and may still provide accurate results if the MP assumptions hold. A reconciliation with lowest cost can be obtained using a dynamic programming framework, in which every gene node is visited once. Under a different event model, the problem may have different complexity, or even be NP-hard.

### 2.1.3. Event Models

The event model determines how incongruence can be interpreted. The three main types of events in a reconciliation are: duplications ($D$), transfers ($T$), and losses ($L$). Any combination of these three events can be included in the event model. Common event models are the DTL, DT, DL, and TL models. Different models are applicable to different data depending on how genes are evolving. The most appropriate event model depends on the biological context.

#### 2.1.3.1. DL Model

The simplest event model considers duplications and losses (DL). This problem was the first recognized event model for reconciliation and was formalized by Page (1994), in which a mapping technique was proposed. The DL reconciliation problem has been approached from several optimization aspects: minimizing the number of events (Durand et al., 2006b; Goodman et al., 1979; Page, 1994), minimizing the size of the reconciled tree (Bonizzoni et al., 2005), and minimizing duplication then loss (Chauve et al., 2008). These measurements of the DL reconciliation infer the same number of duplications but differs in how they define the measurements for unobserved genes (losses). It was later shown that these measures are

actually equivalent (Eulenstein and Vingron, 1998). The optimal reconciliation result can be calculated in linear time (Chauve et al., 2008), and for a given pair of gene and species trees, there is only one optimal solution (Bonizzoni et al., 2005).

#### 2.1.3.2. Inferring Transfer Introduces Two Challenges

Event models considering transfer significantly increase the complexity of reconciliation as two challenges are introduced: multiple optimal solutions and temporal feasibility.

**2.1.3.2.1. Multiple Optimal Solutions** As transfers reduce the restrictions for assigning species labels on each gene node, the gene tree is no longer confined to the structure of the species tree. Gene tree nodes that are closely related may be labeled with species that are distantly related. Further, since both gene duplication and gene transfer increase the number of gene copies, gene copy difference may be explained by multiple combinations of events. This suggests that, from a species tree perspective, if a single species node is the label on multiple gene nodes, the extra copies of the gene may come from duplications in the ancestor, or transfers in from different lineages. Thus, the reconciliation may result in multiple mapping $\mathcal{M}$ tables and among them, the lowest score for mapping the $\rho(T_G)$ to a species node is the same (Figure 2.1). In another word, the minimum cost solution may arrive through different series of events. In fact, the theoretical number of solutions of a gene tree is only loosely bounded under $|V_S|^{|V_G|}$. Because every gene node can have $|V_S|$ number of species labels, and if the cost for these labels are the same, there are $|V_S|$ number of solutions for the gene node. Therefore, the total number of minimum cost solutions is only bound under $|V_S|^{|V_G|}$.

This problem extends to trees generated from real sequence data sets. Typically gene trees are not likely to have exponentially high numbers of solutions; however it is not impossible for large gene trees (with at least 20 leaves) to have over 1 million optimal solutions. Some gene trees have so many solutions that even counting the total number of solutions is expensive in practice. On one hand, as there is no evidence to suggest some solutions are more preferred than others, all the solutions may be equally important and should be considered. On the other hand, when a gene tree has too many solutions, it may be impossible to consider all and the solutions may conflict, leaving little information to consider.

Alternate strategies to cope with the multiple solutions problem exist, and many software packages use different strategies. Xscape counts the number of solutions for different combinations of event numbers and displays them in a picture (Libeskind-Hadas et al., 2014); Eucalypt efficiently enumerates all possible solutions and outputs a subset based on user input (Donati et al., 2015b); Notung collects all solutions and can output all reconciled trees or only a subset of them based on user input; Notung also implicitly enumerates all solutions to calculate summary information on each gene tree reconciliation, such as the average number of transfers, duplications, and losses (Stolzer et al., 2012b). In order to capture the information from multiple solutions and while also not get stuck with gene trees with millions of solutions, Bansal et al. (2013) proposed an approach to randomly sample solutions from all optimal solutions and use the samples as a representation of all solutions. The complexity of random sampling a solution is $O(|V_G||V_S|^2)$, which is greater than regular enumeration implemented in Eucalypt and Notung ($O(|V_G|)$, but the sample of solutions was shown to capture the "signature" of the population of all solutions. However, this described approach was only proposed and tested as a theoretical algorithm, and not implemented in a mature software package available to the public. In general, to record all optimal solutions, the software must perform additional book-keeping and is often slower than software that only generates one solution. For example, the complexity for a DTL algorithm that generates only one solution is $O(|V_S||V_G|)$ (Bansal et al. (2012)). To generate multiple solutions, for each gene node, $O(|V_G|^2)$ is needed to complete a full cost table; therefore, the complexity of an algorithm that can generates all solutions $O(|V_S||V_G|^2)$ (Stolzer et al. (2012b)).

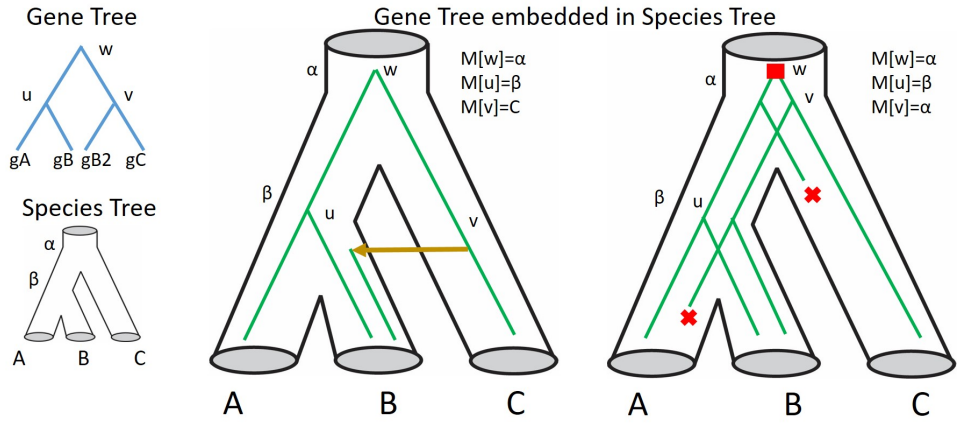



Figure 2.1: Multiple solutions can explain the same incongruence. Left reconciliation is the result of a transfer event, right reconciliation is the result of a duplication and two losses. Depending on $C_T, C_D, C_L$, different reconciliation is preferred.

**2.1.3.2.2. Temporal Consistency** The species tree has implicit temporal constraints on all the species: ancestral species nodes must pre-date their descendants. Similarly, the gene tree implies that species associated with a gene cannot have been younger than the species associated with the gene's descendants. Transfer events introduce extra temporal constraints, because a transfer suggests that the donor species and recipient species must have co-existed at some time point. This temporal constraint between species created by transfer events may conflict with constraints from the species tree, gene tree, and/or from other transfers.

A reconciliation solution is valid only if there exists an ordering of species that satisfies all temporal constraints from the gene tree, species tree, and the inferred transfers. However, if an ordering does not exist, the temporal constraints conflict, and we call the solution "temporally infeasible" (Figure 2.2). These reconciliations must be excluded from the reported solutions, since they cannot represent the actual evolutionary history of a gene family.

For a dated species tree, the transfers inferred always satisfy the temporal constraints, because species that co-exist can be determined from the species tree. This guarantees that every solution generated is temporally feasible. However, for an undated species tree, there are no user supplied constraints to prevent the generation of temporally infeasible solutions during the reconciliation, aside from prohibiting transfers from an ancestor to a descendant. Finding a reconciliation solution that is both valid and at lowest cost for any gene tree and undated species tree is proven to be NP-complete under the TL model (Hill et al., 2010) and DTL model (Hallett et al., 2004; Ovadia et al., 2011; Tofigh et al., 2011). The only guaranteed way to find a solution is through exhaustive search. However, finding a lowest cost solution, disregarding its temporal validity, only takes polynomial time. Therefore, a reasonable heuristic for this problem is to generate candidate solutions, then check for temporal consistency. If a valid solution is among these candidate solutions, then it is also an optimal solution. However, if no valid solutions were found in these candidate solutions, then the only option to find an optimal solution is by exhaustive search with exponential complexity.

### 2.1.3.3. TL Model

Under the TL event model, a transfer can be represented by a Subtree Prune and Regraft (SPR) operation acted on a species tree. The problem of finding the minimum number of SPR operations is NP-hard but can be reduced to fixed parameter tractable when parameterized

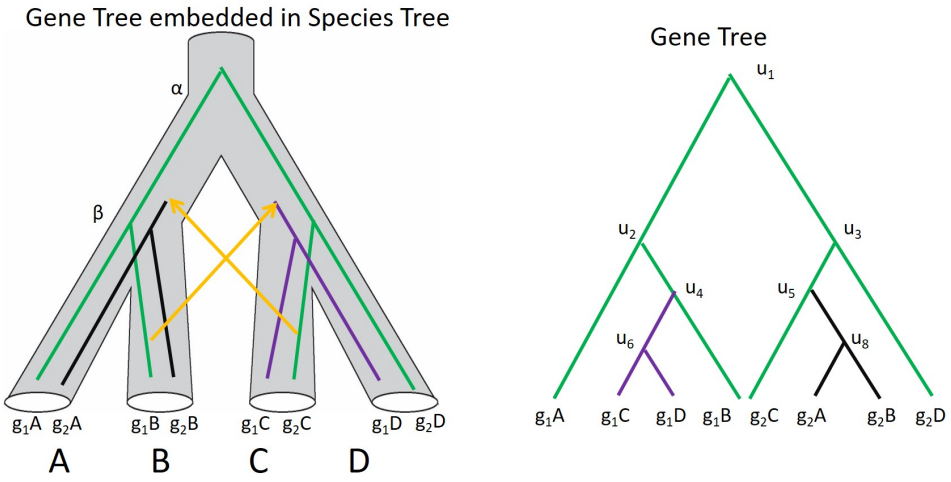



Figure 2.2: A reconciliation solution involving two transfers. The species tree demonstrates two transfers that are conflict with each other. It is impossible to find timestamps for species node $\beta$ and $\gamma$ that satisfies the timing constraints from the transfers, leading to an invalid solution

by the distance between the gene tree and species tree (Hill et al., 2010). Several approaches were developed based on this definition, including a heuristic polynomial-time approach based on bipartition dissimilarity that is best fit for small number of transfers (Boc et al., 2010), a generic heuristic polynomial-time algorithm that handles HGT without restrictions (Nakhleh et al., 2005) and an exact algorithm when tree distance is parameterized (Hill et al., 2010).

#### 2.1.3.4. DTL Model

The DTL event model is the full event model that involves inferring all three common gene events. In the context of gene tree and species tree reconciliation, algorithms have been developed to find DTL reconciliation solutions with a dated binary (Doyon et al., 2010), undated binary (Bansal et al., 2012; Tofigh et al., 2011), or undated non-binary (Stolzer et al., 2012b) species tree using a polynomial time algorithm. When timing is assigned to every node of the species tree, contemporaneous branches are known explicitly. In this scenario, DTL reconciliation can be solved in polynomial time (reviewed by Doyon et al. (2011); Huson and Scornavacca (2011)). However, this more restricted model requires estimation of speciation times, which may not be available. An exact algorithm for DT reconciliation has been presented by Tofigh et al. (2011). It is mainly conceptual and may

not be practical due to its exponential complexity. Tofigh et al. (2011) also provided a heuristic algorithm to find the minimum cost reconciliation solution for the DT event model, and suggested using the number of losses as a secondary criteria to select from the multiple candidate solutions. Bansal et al. (2012) were able to further reduce the complexity for the DTL heuristic algorithm to linear time, however it only generates one event history, making it more suitable for other applications that only require knowing the minimal cost (Section 2.2). To test for temporal inconsistency, Tofigh et al. (2011) proposed a validation procedure based on topological sorting for the DT event model and Stolzer et al. (2012b) showed that including loss events requires additional constraints on the DT validation procedure and introduced the validation procedure for the DTL event model.

#### 2.1.3.5. Host-Parasite: Species Event Model

The event models involving transfer are also closely associated with a different problem that considers a group of parasites and a group of hosts (Charleston, 1998), in which, a "gene transfer" is analogous to a host-parasite switching event; a gene divergence due to speciation is analogous to "co-speciation" between parasite and host. This similar problem aims to reconcile a "parasite tree" with its "host tree" and infer host-parasite switching, co-evolution, and duplication event and was also proved to be NP-complete (Libeskind-Hadas and Charleston, 2009; Ovadia et al., 2011). Conceptually, all event models are applicable to the host-parasite input data. However, several approaches specific to host-parasite have been developed. One major difference between the host-parasite problem and the gene-species reconciliation problem is that co-speciation (analogous to speciation in a gene tree) may also be assigned a weight in host-parasite reconciliation. In gene-species reconciliation, a speciation event usually has zero cost. Under the host-parasite paradigm, some problems consider only host-parasite switching events and parasite extinction events, (analogous to the TL event model). This problem was tackled by Ronquist (1995), who developed a general parsimony approach that minimizes the total weighted sum of events. A Bayesian approach was also developed for the "host-parasite" switching event under a probabilistic framework and is claimed to be more robust against phylogenetic error (Huelsenbeck et al., 2000). Other models considers host-parasite switching, co-evolution, and duplication. Several algorithms were also developed to solve this problem using a DTL reconciliation algorithm with slight modification. The Jane (Conow et al., 2010) software tool combines dynamic programming and a genetic algorithm as a heuristic to find a reconciliation solution while allowing the

user to limit host-switch (analogous to transfer) distance and to assign different costs in different regions of the tree. The CoRe-PA (Merkle and Middendorf, 2005) software tool utilizes a parameter-adaptive approach in the dynamic programming framework so that it is not required for the user to assign a specific cost for each event.

## 2.2. Event Inference in Other Applications

The reconciliation problem under a specified optimization criteria is also a way to measure the gene tree. The parsimony score provides a means of scoring a gene tree based on its incongruence with the species tree. This scoring mechanism is therefore also useful as an intermediate step in solving several other problems, such as inferring a species tree from a set of gene trees, resolving non-binary gene trees, and rooting an unrooted gene tree.

### 2.2.1. Species Tree inference

**Problem Statement** Given $k$ rooted gene trees, and the species from which the genes are sampled, find a species tree $T_S$ topology such that the sum of scores for reconciling each gene tree with the species tree is minimal.

Note that as the goal of this problem is to find a species tree topology, it is not required to find the full event history for each gene tree and a reconciliation score is sufficient. Under DL the event model, this species tree inference problem is NP-hard (Bonizzoni et al., 2005), and a graphic formulation was proposed by Mirkin et al. (1995); Wehe et al. (2008).

### 2.2.2. Rooting and Resolving a Gene Tree

**Problem Statement: Rooting**
Given an unrooted gene tree and a rooted species tree, find the optimal root of the gene tree that results in the minimum cost in the reconciliation with the species tree under a specified event model.

An unrooted gene tree with $k$ leaves has $k$ branches. In the rooting problem, every branch in the gene tree is an appropriate candidate root. Then, finding the best root of the gene tree

can be achieved by iterating over every branch as the new root, and comparing the result of the reconciliation of the rooted gene tree with the species tree.

**Resolving: Problem Statement**
Given a rooted gene tree with non-binary gene nodes and a rooted species tree, find the best binary resolution(s) for each non-binary gene node and reconstruct a new gene tree, such that the cost of reconciliation is minimized under a specified event model.

For a non-binary gene node of size $k$, the number of binary resolutions is the double factorial $(2k-3)!!$. This number increases exponentially as the size of the non-binary gene node increases. Thus, simply enumerating all possible binary resolutions is not practical. Under the DL event model, the optimal binary resolution can be solved in linear time (Lafond et al., 2016; M. et al., 2012). However, under the DTL event model, there does not exist any fast algorithm to guarantee an optimal binary resolution in polynomial time. Instead, many heuristic algorithms have been developed to sample the solution space with modified tree modification techniques (NNI, SPR) (Bansal et al., 2015; Nguyen et al., 2013). In Chapter 4, I describe the DTL resolve problem in detail and developed a heuristic algorithm that shows promising results.

## 2.3. General Algorithm for Reconciliation

Many algorithms have been developed for the purpose of solving the reconciliation problem. Each one follows a similar dynamic programming framework and infers the gene events using a very similar mapping technique. The general algorithm for inferring gene events is to find a mapping function ($\mathcal{M} : g \in V_G \rightarrow s \in V_S$) that maps every node in the gene tree onto a node in the species tree. Biologically, assigning a species node label to a gene node indicates that the species genome contains this gene. For the purpose of this description, I define the gene tree as $T_G = (V_G, E_G)$ and the species tree as $T_S = (V_S, E_S)$, where $V_i$ represents the set of tree vertices and $E_i$ represents the edges in tree $T_i, i \in \{G, S\}$. Given two nodes $u, v$ from the same tree $T_i$, if $u$ is on the path from $v$ to the root of $T_i$, then $u$ is an ancestor of $v$, designated $u \geq_i v$, and $v$ is a descendant of $u$, designated $v \leq_i u$. If $v \not\geq_i u$ and $u \not\geq_i v$, $u$ and $v$ are incomparable, designated $u \lneqq_i v$. Given a gene node $g \in V_G$ mapped to species node $s \in V_S$, $s$ is called the label of $g$. This information can be stored in a mapping table $\mathcal{M}$ of size $|V_G|$, where $\mathcal{M}[g] = s$ (Figure 2.3). Note that the internal nodes in species trees and

gene trees represent ancestral species and ancestral genes; they are also considered in the mapping table $\mathcal{M}$. After $\mathcal{M}[g]$ is defined for every $g \in V_G$, the gene events can be inferred by analysing the species labels on $g$ and its children for every ancestral gene node. For example, given a gene node $g$, if the species labels of the children, $\mathcal{M}[l(g)]$ and $\mathcal{M}[r(g)]$, are the same as the species label of g, the gene, in species $\mathcal{M}[g]$, was duplicated, resulting in two copies of the gene in the same species; a duplication event is assigned to have occurred at $g$. The loss event(s) that occurred along a gene branch are inferred by comparing the species labels of $p(g)$ and $g$. The number of losses is easily calculated as the difference in heights between the two species labels: $\mathcal{M}[p(g)]$ and $\mathcal{M}[g]$ in $T_S$. Intuitively, the species in which losses occurred are inferred based on the species that are missing between the two species nodes in $T_S$.

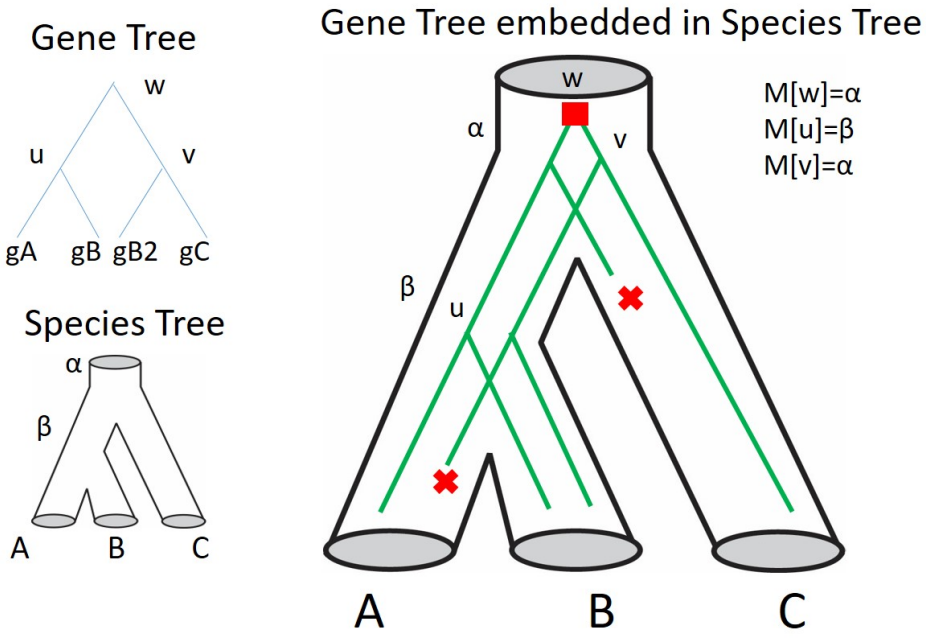



Figure 2.3: Reconciliation of a gene tree with a species tree. The solution is represented as $\mathcal{M}$ for every gene node.

In this section, I will first describe the algorithm for the DL event model. Then I will discuss the algorithm for the DTL event model and how the two challenges with transfer events are approached in the algorithm. The general algorithms described here are demonstrations to show the general framework of the problem. Many of the studies described in Section 2.1.3 use the same framework, but differ by the definition of gene events and the rules of how events occur.

### 2.3.1. DL Model

The reconciliation problem under the DL event model is formally defined:
**Input:** $T_G, T_S$, DL event model, and $\mathcal{M}[g] \forall g \in L(T_G)$
**Output:** $\mathcal{M}[g] \forall g \in V_G$.
Recall that for the parsimony criterion, $\mathcal{M}$ is inferred by minimizing the number of duplications and losses. Optionally, an event table $\mathcal{E}[g] \forall g \in V_G$ where $\mathcal{E}[g] \in \{D, S\}$ may also be included.

In the input, the leaves of the gene are assigned the species node that represents the species from which these genes are sampled. The algorithm to assign the species labels for ancestral gene nodes is fairly straightforward. The best mapping for a gene node is the least common ancestor (LCA) of the two species labels of its children, where the LCA of two nodes is the lowest node (i.e., furthest from the root) in the tree that is the ancestor of both nodes. A post-order traversal of the gene tree can visit every gene node once and assign the best species node label for that gene node (Algorithm 2). A pre-processing step simplifies this procedure by calculating the LCA for every pair of species (see Algorithm 1) and storing them in a matrix of size $|V_S| \times |V_S|$. The LCA table is populated recursively, without repetition. This process can be completed in polynomial time with the size of species node set ($O(|V_S|^2)$). Then, determining the LCA at a gene node requires only to look up the value in this table corresponding to the species labels of the gene nodes' children. Given the LCA table, this process can be completed in polynomial time ($O(|V_G|)$) with size of $T_G$ (Algorithm 2).

**Algorithm 1** Precompute LCA table

```
1 lcatable={}
2 decendants={} # imporper decendants, include self
3 BuildLCATable(T_S) {
4    for each s ∈ V_S \ L(T_S) post order{
5      for each s_1, s_2 ∈ decendants[l(s)], decendants[r(s)]{
6        lcatable[s_1][s_2]=s
7      }
8      decendants[s].extend(decendants[l(s)],decendants[r(s)],s)
9    }
10 }
```

**Algorithm 2** DL Reconciliation

```
1 # M contains mapping for all L(T_G)
2 DLreconciliation(T_G) {
3     for each g ∈ V_G \ L(T_G) {
4         M[g] = LCA(M[l(g)], M[r(g)])
5     }
6 }
```

### 2.3.2. DTL Model

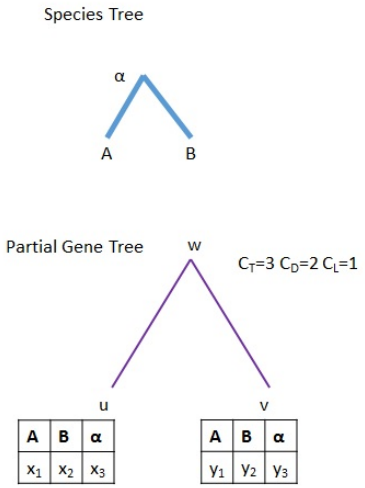
Here I provide a general strategy for DTL reconciliation, developed for a binary gene tree and a binary species tree under the parsimony criterion.

**Input:**$T_G, T_S$, DTL event model, and $\mathcal{M}[g] \forall g \in L(T_G)$
**Output:**$\mathcal{M}[g] \forall g \in V_G$.
Recall that for the parsimony criterion, $\mathcal{M}$ is inferred by minimizing the number of duplications, transfer and losses. Optionally, an event table $\mathcal{E}[g] \forall g \in V_G$ where $\mathcal{E}[g] \in \{D, T, S\}$ may also be included.

Because of the two major challenges involved in this event model (multiple optimal solution and temporal feasibility discussed in Section 2.1.3.2), the general reconciliation under the DTL event model usually consists of two steps: 1) generating reconciliations with minimum cost, and 2) filtering out the reconciliations that are temporally infeasible. Here I clarify some terms that I use regarding temporal feasibility and the results of steps 1 and 2. A *candidate* solution is a solution that has minimal cost, a *valid* solution is a solution that is temporally feasible, and an *optimal* solution is a solution that both has minimal cost and is temporally feasible (i.e. is a candidate solution and also a valid solution). Step 1 generates candidate solutions. Step 2 tests these candidate solutions and reports any optimal solutions.

In addition to the two challenges, the reconciliation problem with the DTL event model is much more complicated than with the DL event model. Because of transfers, the species label of a parent gene may no longer be an ancestor of the species label of the gene. The mapping of the parent can be (almost) any species node in the species tree. This makes the LCA mapping strategy used in DL obsolete. Instead, all possible species mappings must be considered. In addition, the best $\mathcal{M}[g]$ for a subtree of the gene tree is not necessarily the best $\mathcal{M}[g]$ for the entire gene tree. Therefore, it is impossible to decide the best $\mathcal{M}[g]$ for a gene node until all gene nodes above and below $g$ are visited.

Therefore, DTL reconciliation proceeds using a dynamic programming approach. Given gene tree $T_G$ and a species tree $T_S$, we define two cost tables: $\mathcal{K}$ saves the cost of mapping every gene node to every species node, and $\mathcal{H}$ stores the necessary bookkeeping information that resulted in the corresponding cost. The entry for mapping gene node $g$ to species node $s$ is denoted $\mathcal{K}_g[s]$. Note that $\mathcal{K}$ is a two-dimensional table, and $\mathcal{K}_g$ is a one-dimension array. The cost table entries for a gene node only depend on the cost arrays of the two children nodes of $g$ (Figure 2.4). Calculating entries in $\mathcal{K}$ and $\mathcal{H}$ and generating a candidate solution require a two step dynamic programming procedure. The first step is to calculate, in post-order, all "potential" best mappings for every gene node and store the cost in $\mathcal{K}$ and the corresponding information in $\mathcal{H}$. The second step is to traceback through the cost tables from the root of gene tree to generate the best mappings for the entire gene tree. Here I provide a basic algorithm for reconciliation under the DTL model that calculates the minimum cost. Then I will also show how this basic algorithm can be sped up by neglecting combinations of species nodes that are known to never be parsimonious and how it can be modified to generate all candidate solutions.




Intermediate Table for w

| Left, Right Label | Base Cost | M[w]=A | M[w]=B | M[w]=α |
|---|---|---|---|---|
| A, A | $x_1+y_1$ | D,2 | 2T,6 | D+2L,4 |
| A, B | $x_1+y_2$ | T,3 | T,3 | S,0 |
| A, α | $x_1+y_3$ | ∞ | ∞ | D+L, 3 |
| B, A | $x_2+y_1$ | T,3 | T,3 | S,0 |
| B, B | $x_2+y_2$ | 2T,6 | D,2 | D+2L,4 |
| B, α | $x_2+y_3$ | ∞ | ∞ | D+L,3 |
| α, A | $x_3+y_1$ | ∞ | ∞ | D+L,3 |
| α, B | $x_3+y_2$ | ∞ | ∞ | D+L,3 |
| α, α | $x_3+y_3$ | ∞ | ∞ | D,2 |

Cost Table for w

| A | α | B |
|---|---|---|
| D, $x_1+y_1+2$,A,A | S, $x_1+y_2$,A,B | D, $x_2+y_2+2$,B,B |

Figure 2.4: Cost table for $w$ calculated from iterating all possible labels for $u$, labels for $v$ and labels for $w$. For example, if $\mathcal{M}[u] = A, \mathcal{M}[v] = A$, then the cost for $\mathcal{M}[w] = A$ is 2 because these three labels indicate a duplication event has occurred. The total cost to map $w$ in $A$ in this situation is $x_1 + y_1 + 2$. Among all rows, for each label for $w$, only the situation that costs the minimum is necessary to be saved. Assuming the entry with a red circle has the minimum cost. The Cost Table for $w$ is constructed by only saving the minimum cost rows and converting to a tuple that saves the event, total cost, and left, right label.

#### 2.3.2.1. Step 1: Calculating Cost Table

Generating cost tables for every gene node follows a dynamic programming framework, where each gene node $g$ is visited post-order, and $\mathcal{K}_g, \mathcal{H}_g$ are calculated during each visit. Given a gene node $g$, and its children $l(g), r(g)$, the events and cost at $g$ are computed based on the combination of species assigned to its children. Therefore, an intuitive approach to calculate $\mathcal{K}[g]$ is to exhaustively try all combinations of species labels for gene node $g$, its left child ($l(g)$) and right child ($r(g)$). The total number of such possible combinations for any gene node is $|V_S| \times |V_S| \times |V_S|$. Thus, this algorithm has complexity of $O(|V_G||V_S|^3)$ (see Algorithm 3). Given a combination of three species nodes $s, s_1, s_2$, which are assigned as species labels for $g, l(g), r(g)$, the rule for calculating the cost works as follows (also see Algorithm 4):

- If $s$ is not an ancestor of either $s_1$ or $s_2$, then two transfers must have occurred, one from $s$ to $s_1$, and one from $s$ to $s_2$.

- If $s_1 \not\leq_S s_2$, then either a transfer or speciation event must have occurred. If $s$ is a common ancestor of both $s_1$ and $s_2$, then a speciation occurred at $g$, and losses are inferred between (1) $s$ and $LCA(s_1, s_2)$, (2) $LCA(s_1, s_2)$ and $s_1$, and (3) $LCA(s_1, s_2)$ and $s_2$. If $s$ is not a common ancestor of $s_1$ and $s_2$, then a transfer occurred. If $s$ is an ancestor of $s_1$, then a transfer occurred from $s$ to $s_2$ and losses are inferred between $s$ and $s_1$. The inverse is true for the case where $s$ is ancestor for $s_2$.

- If $s_1$ is comparable to $s_2$, then $s_1$ is an (improper) ancestor of $s_2$ or $s_2$ is an (improper) ancestor of $s_1$ and $s$ can only be the ancestor of both $s_1$ and $s_2$. A duplication must have occurred. If $s_1$ is an (improper) ancestor of $s_2$, the duplication is occurred in $s_1$ and losses are inferred between $s$ and $s_1$ and between $s_1$ and $s_2$. The inverse is true for the case where $s_2$ is an (improper) ancestor of $s_1$.

Typically case 1 is not considered to be parsimonious, thus for a given gene node $g$, and the species labels on left and right children $s_1 = \mathcal{M}[l(g)], s_2 = \mathcal{M}[r(g)]$, the species that are incomparable to both $s_1, s_2$ are not necessary to be considered when calculating $\mathcal{K}$ for $g$. This potentially reduces the number of species label combinations and speeds up the algorithm. In addition, if $g$ can be mapped to a species node that is common ancestor of $s_1$ and $s_2$, with some cost, then mapping $g$ to ancestors of $s$ only increases the number of losses accordingly. These ancestors of $s$ are not necessary to be considered in the $\mathcal{K}_g$, because they will be considered when the cost table for $p(g)$ is computed. In fact, to generate $\mathcal{K}$ for a gene node $g$,

**Algorithm 3** DTL reconciliation that iterate all possible labels.

```
1 DTLreconcile(V_G, V_S) {
2    for each g ∈ V_G post order{
3      for each s_1 ∈ V_S {
4        for each s_2 ∈ V_S {
5          for each s_3 ∈ V_S {
6            cost=calculateCost(g, s_1, s_2, s_3)
7            K_g[s_3] = cost if cost is less than K_g[s_3])
8          }
9        }
10     }
11   }
12 }
```

**Algorithm 4** Calculating Cost for One $g \in V_G$

```
1 calculateCost(g, s_1, s_2, s_3) {
2    # Invalid
3    if s_1 >_S s or s_2 >_S s:  return ∞
4    lca=LCA(s_1, s_2)
5    leftCost=K[l(g)][s_1], rightCost=K[r(g)][s_2]
6    if s ⋡_S s_1 and s ⋡_S s_2:  return 2c_τ+leftCost+rightCost;
7    if s_1 ⋡_S s_2 {
8      # speciation
9      if s ≥_S lca {
10       N_L = h(s) − h(lca) − 1 + h(lca) − h(s_1) − 1 + h(lca) − h(s_2) − 1
11       return C_λ N_L+leftCost+rightCost
12     }
13     # Transfer
14     if s ≥_S s_1 { # Transfer
15       N_L = h(s) − h(s_1) − 1
16       return C_τ + C_λ N_L+leftCost+rightCost
17     }else{
18       N_L = h(s) − h(s_2) − 1
19       return C_τ + C_λ N_L+leftCost+rightCost
20     }
21   }
22   # Duplication
23   if s_1 > s_2 {
24     N_L = h(s_1) − h(s_2) + h(s) − h(s_1) − 1
25     return C_δ + C_λ N_L+leftCost+rightCost
26   }
27   N_L = h(s_2) − h(s_1) + h(s) − h(s_2) − 1
28   return C_δ + C_λ N_L+leftCost+rightCost
29 }
```

only the species labels on $l(g)$ and $r(g)$ need to be enumerated (Stolzer et al., 2012b; Tofigh, 2009)(see Algorithm 5). For each pair of species labels $(s_1, s_2)$ on $l(g)$ and $r(g)$, the species labels that need to be considered to map to $g$ in $\mathcal{K}_g[s]$ are $\{s_1, s_2, LCA(s_1, s_2)\}$ for generating one solution only, and $\{s|(s >_S s_1 \text{ or } s >_S s_2)\ \&\ s \leq_S LCA(s_1, s_2)\}$ for generating all candidate solutions. This effectively reduces the complexity to $O(|V_G||V_S|^2)$, or $(O(|V_G||V_S|^2 h)$ for all reconciliations) and still manages to generate lowest cost solution(s). The complexity can be further reduced to $O(|V_G||V_S|)$ if only one solution is required regardless of temporal feasibility (Bansal et al., 2012).

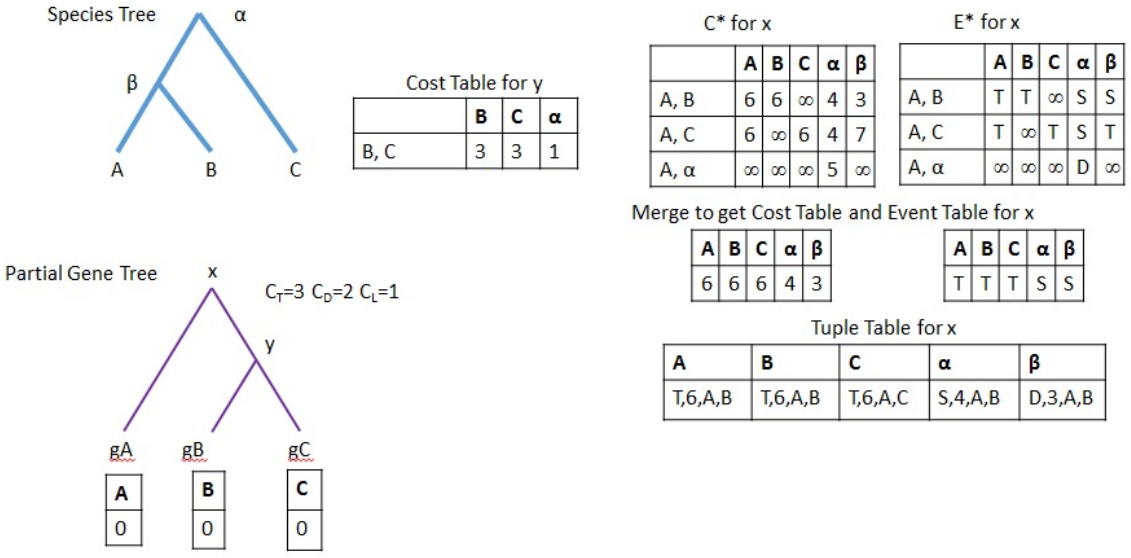



Figure 2.5: Dynamic programming to calculate cost table for a binary gene node $x$. For a binary node, the procedure tries all possible ways to map the children gene node with different species nodes and determine the cost of map $x$ to different species nodes. The merge process throws out scenarios that generate non-optimal costs and generate a tuple table, in which, each cell stores the event, total cost, left child label, and right child label.

Saving just the minimum cost for mapping a gene node to a species node in $\mathcal{K}$ is useful when the algorithm is used to score gene trees in species tree inference or a rooting algorithm. However, for more comprehensive results, such as generating the full event history, the mapping table $\mathcal{M}$ for every gene node is required. To generate the complete mapping of every gene node, some extra information is required to be stored when calculating entries in $\mathcal{K}$. This information is stored in the corresponding table $\mathcal{H}$:

- For a gene node $g$, and any species node $s$, the minimum cost is stored in $\mathcal{K}_g[s]$. This minimum cost is calculated for one of the many combinations of species labels $(s_1, s_2)$

on $l(g)$ and $r(g)$. However, to correctly track the mapping of $g$ and its children, the species labels on the children of $g$, $s_1$ & $s_2$, that resulted in this cost are also required to be stored. For this purpose, one could construct a table with the same dimension as $\mathcal{K}$ to store $s_1$ and $s_2$ (i.e. the event table in Figure 2.5). This table is denoted $\mathcal{H}$. Therefore, $\mathcal{H}_g[s]$ should be at least a tuple containing the label of $l(g)$ and the label of $r(g)$ (Eq. 2.2). More information could be saved in the entry to avoid redetermining the events, such as the event that occurred at the gene node and any event that occurred on the branches $< g, l(g) >$ and $< g, r(g) >$. This information could save a lot of runtime in the later traceback process (Secion 3.1.2).

$$\mathcal{K}_g[s] = cost \tag{2.2}$$

$$\mathcal{H}_g[s] = (\mathcal{M}(l(g)), \mathcal{M}(r(g))) \tag{2.3}$$

- To generate multiple candidate solutions, the situation where *cost* is the same as existing value in $\mathcal{K}$ should also be saved. Therefore, we further expand the $\mathcal{H}_g[s]$ for any $g \in V_G, s \in V_S$ to store a list of tuples instead of single tuple as described previously (see Algorithm 6).

- A traceback algorithm to traverse the gene tree pre-order and assign $\mathcal{M}(g)$ based on the cost table of the parent $\mathcal{K}_{p(g)}$, except for $\rho$, where $\mathcal{M}(\rho)$ is the entry where $\mathcal{K}_\rho[s]$ has the minimum costs. During this process, as the $\mathcal{M}$ is calculated, the $\mathcal{E}$ table can also be calculated for every gene node. Unlike speciation and duplication, where gene events occur on one gene node, a transfer event occurs on a gene branch. A transfer on branch $< p(g), g >$ is associated with two gene nodes: the donor $p(g)$ and the recipient $g$. As a gene node has two children and only one parent, assigning "T" to $\mathcal{E}[p(g)]$ is ambiguous because the recipient could be either of the children. Therefore, $\mathcal{E}[g] =$ "$T$" if $g$ is a recipient of a transfer event.

Once every node in the gene tree has been visited and $\mathcal{K}$ and $\mathcal{H}$ have been generated, the lowest cost solution can be found as the mapping that results in lowest cost at the root $\rho(T_G)$:

$$mincost = min(\mathcal{K}_{\rho_G}) \tag{2.4}$$

In Section 3.1, I describe the algorithm (and provide pseudocode) that we developed to enumerate multiple candidate solutions using a modified backtracking algorithm.

**Algorithm 5** DTL algorithm only Iterate Labels on $C(g)$

```
1 DTLReconcile(V_G, V_S) {
2    for each g ∈ V_G post order{
3      for each s_1 ∈ V_S {
4        for each s_2 ∈ V_S {
5          generateCostTable(g, s_1, s_2)
6        }
7      }
8    }
9 }
10
11 generateCostTable(g, s_1, s_2) {
12   leftCost = K[l(g)][s_1], rightCost = K[r(g)][s_2]
13   # Duplication or speciation
14   lca = LCA(s_1, s_2)
15   if s_1 ⋦_S s_2 {
16     # Speciation
17     N_L = h(lca) − h(s_1) − 1 + h(lca) − h(s_2) − 1
18     cost = C_λ N_L + leftCost + rightCost
19     K[g][lca] = cost if cost is less than K[g][lca]
20     # Transfer
21     cost = C_τ
22     K[g][s_1] = cost if cost is less than K[g][s_1]
23     K[g][s_2] = cost if cost is less than K[g][s_2]
24   }else{
25     # Duplication
26     N_L = |h(s_2) − h(s_1)|
27     cost = C_δ + C_λ N_L + leftCost + rightCost
28     K[g][lca] = cost if cost is less than K[g][lca]
29   }
30 }
```

**Algorithm 6** Calculating Cost for $g \in V_G$ and update $\mathcal{K}$

```
1  generateCostTable(g, s_1, s_2) {
2     leftCost = K_{l(g)}[s_1], rightCost = K_{r(g)}[s_2]
3     # Duplication or speciation
4     lca = LCA(s_1, s_2)
5     if s_1 ≰_S s_2 {
6       # Speciation
7       N_L = h(lca) − h(s_1) − 1 + h(lca) − h(s_2) − 1
8       cost = C_λ N_L + leftCost + rightCost
9       update(g, lca, s_1, s_2, cost, "S")
10      # Transfer
11      cost = C_τ
12      update(g, s_1, s_1, s_2, cost, "T")
13      update(g, s_2, s_1, s_2, cost, "T")
14    }else{
15      # Duplication
16      N_L = |h(s_2) − h(s_1)|
17      cost = C_δ + C_λ N_L + leftCost + rightCost
18      update(g, lca, s_1, s_2, cost, "D")
19    }
20 }
21
22 update(g, s, s_1, s_2, cost, ϵ){# ϵ represent the event
23    if cost < K_g[s] or K_g[s] is empty:  H_g[s] = [(s_1, s_2, ϵ)]
24    if cost = K_g[s]:  H_g[s].append((s_1, s_2, ϵ))
25 }
```

#### 2.3.2.2. Step 2: Traceback to Generate Solution

After the two cost tables are generated, the next step is to generate the appropriate $\mathcal{M}$ for all gene nodes while also collecting gene events and total cost. For any gene node $g$, and possible label $s$, the entry in $\mathcal{K}_g[s]$ represents the lowest cost to map $g$ to $s$. This is equivalent to reconciling the subtree $T_G(g)$ with $T_S$ and $\mathcal{M}[g] = s$. Therefore, the lowest cost solution (a candidate solution), can be obtained by mapping $\rho_G$ to any species node that yields lowest cost in $\mathcal{K}_g$.

The detailed algorithm works as follows: First, $\mathcal{M}(\rho_G)$ is set to $s*$, such that:

$$s* = \underset{s \in V_S}{\operatorname{argmin}}\{\mathcal{K}_{\rho_G}[s]\}. \tag{2.5}$$

Identifying this label at $\rho_G$ initiates the traceback procedure. A tuple of information, including left best label ($s_1$) and right best label ($s_2$), is saved in the entry $\mathcal{H}_{\rho_G}[s]$. Recall that a given entry in $\mathcal{H}_{\rho_G}[s]$ specifies the $\mathcal{M}[l(\rho_G)]$ and $\mathcal{M}[r(\rho_G)]$ that resulted in an optimal reconciliation of $T_G(\rho_G)$ with $T_S$ when $\mathcal{M}[\rho_G] = s$. Therefore, given an entry for gene node $g$, we can determine the $\mathcal{M}$ for the left and right children. The left best label and right best label $(s_1, s_2)$ are assigned to the $\mathcal{M}[l(\rho_G)]$, and $\mathcal{M}[r(\rho_G)]$. Then we use the information in $\mathcal{H}_{l(\rho_G)}[\mathcal{M}[l(\rho_G)]]$ to assign the species node mapping to the next two children. By recursively repeating this procedure, the full mapping table ($\mathcal{M}$) for every gene node is produced. Then the corresponding gene events can be calculated, and the original gene tree $T_G$ can be converted to a reconciled gene tree ($T_G^*$) (Algorithm 7). Once the gene tree is augmented with $\mathcal{M}$ for every gene node, the location of gene duplication, transfer, and losses can also be determined in the gene tree. If the $\mathcal{H}$ collects all entries that yield the same cost, then all candidate solutions can be generated with a backtracking algorithm. The algorithm to generate all solutions are described in Chapter 3.

#### 2.3.2.3. Checking for Temporal Consistency

The candidate solution generated by the two step algorithm is not optimal until its temporal feasibility is validated. To determine if a solution is valid, a timing graph $G_T = (V_T, E_T)$ is constructed (discussed in (Stolzer et al., 2012b; Tofigh, 2009)), where each node represents a node in the species tree, and each edge $< u, v >$ is directional and represents a timing constraint where $u$ should pre-date $v$.

**Algorithm 7** DTL Traceback

```
1 for each (s) ∈ arg min_{s*∈V_S}{K_{ρ_G}[s*]} {
2   enqueue (ρ_G, s) to solutionQueue
3 }
4 while solutionQueue is not empty {
5   dequeue (g, s) from solutionQueue
6   traceback(g, s, T_G)
7   Save M as a candidate solution
8 }

traceback(g, s, T_G) {
9   M(g) = s;
10  dequeue {s_l, s_r, ϵ} from H_g[s]
11  E(g) = ϵ
12  if g ∉ L(T_G) {
13    traceback(l(g), s_l, T_G); traceback(r(g), s_r, T_G)
14    while H_g[s] is not empty {
15      dequeue {ϵ, s_1, s_2} from H_g[s]
16      enqueue (l(g), s_1) to solutionQueue
17      enqueue (r(g), s_2) to solutionQueue
18    }
19  }
20 }
```

The temporal feasibility problem can be reduced to the graphical problem called topological sorting, solvable in $O(|V_G| + e)$, where $e$ represents the number of edges in the timing graph. The edges are added as timing constraints carried by species tree and the inferred transfers. Note that under different models, and different assumptions, the temporal constraints carried by transfers may be different. A timing graph is valid if no cycles are detected; therefore this step is also called cycle checking. After the timing graph is constructed, a topological sorting algorithm is applied to it to validate the temporal feasibility. This step is described in detail in Section 3.2, where I propose two useful techniques to speed up the process. Note that depending on the source of the gene tree, this cycle checking step may not be necessary. For example, Hallett et al. (2004) reported that under the DT event model, no gene tree solutions were found temporally infeasible when they applied the reconciliation algorithm to a simulated dataset. However, as a general approach, where the data is not specified, the validation step should always be in action, at least in our opinion.

# Chapter 3

# Extending DTL Reconciliation

In this chapter, I discuss my contributions in the field of DTL reconciliation. First, I will describe my extension of the DTL reconciliation algorithm that incorporates incongruence caused by ILS. I present the first reconciliation algorithm under a DTL event model while excluding ILS. I also propose a modification to the algorithm as a solution for one of the two major challenges of DTL reconciliation: multiple optimal solution. Next, I highlight two techniques that significantly speed up the runtime for validation of DTL and DTLI reconciliation. Finally, I modify the DTL reconciliation model to infer domain-level events by reconciling a domain and a gene tree along with a species tree. This extension solves reconciliation on three layers, vastly complicating the solution space. I additionally created a heuristic which made solving this problem tractable. My results on the DTLI reconciliation algorithm (Section 3.1) and on domain-gene tree reconciliation (Section 3.3) have been published in two articles (Stolzer et al., 2012b, 2015), which were written with collaborators in the laboratory. In this chapter, I focus on my contribution to those papers, which are the theoretical work on the algorithms.

## 3.1. DTL with incomplete lineage sorting

In general, the input of DTL reconciliation is a binary gene tree and binary species tree. The binary species tree requirement suggests an implicit assumption that the inter-speciation times are long enough for allelic variation to fix so that incomplete lineage sorting (ILS) does

not occur. However, this assumption may not be valid if the species tree branches represent short time spans (i.e. short branches). As the resolution of sequenced species increases, inter-speciation time decreases, and species trees that contain such short branches will become more common. In these short branches, allelic variations can survive through multiple species divergences, resulting in ILS, which in turn may cause the gene tree topology to differ from the species tree. Therefore, regular DTL reconciliation algorithms that do not consider ILS may incorrectly misinterpret the signal of ILS as gene events (i.e. duplication or transfer).

Here, I present the first reconciliation algorithm that infers duplication, transfer, and loss events while also excluding incongruence due to ILS. I refer to this as DTLI reconciliation. In particular, this reconciliation distinguishes between incongruence that could only arise through gene events and incongruence that could be explained by ILS. This algorithm is also the first DTL algorithm that is capable of reconciling a binary gene tree with a non-binary species tree.

Recall that DTL reconciliation proceeds in three steps: generating a cost table, tracing back through the tables to generate solutions, and checking solutions for temporal feasibility. Here, I discuss these steps for DTLI reconciliation in the following sections:

1. Section 3.1.1: applying a model that handles ILS to the dynamic programming framework and generating two cost tables $\mathcal{K}$ and $\mathcal{H}$ for every pair of genes, $g \in V_G$, and species, $s \in V_S$.

2. Section 3.1.2: generating all candidate solutions by traceback through the cost tables and calculating all mappings $\mathcal{M}$ for every $g \in V_G$.

3. Section 3.1.3: checking temporal consistency for each generated solution $\mathcal{M}$.

### 3.1.1. Model to handle ILS and Multiple Solutions

My two major innovations for DTLI reconciliation are:

1. Representing branches in a species tree where with ILS may have occurred.

2. Distinguishing incongruence due to ILS from incongruence caused by gene events.

Here I review these challenges.

#### 3.1.1.1. Representation of ILS as Non-binary Species Tree Node

Under the DTLI model, I reconcile a binary gene tree with a non-binary species tree. A species tree branch represents the inter-speciation times between a species $s$ and its parent $p(s)$. If the time between $s$ and $p(s)$ is long enough that I assume ILS can be eliminated, then I include the branch to indicate that it is not necessary to consider ILS at $p(s)$. However, if the time is too short, such that the speciation events at $p(s)$ and $s$ occurred in rapid succession, it suggests that ILS cannot be ignored. In this case, the branch is collapsed into 0 length. This effectively removes $s$ from the tree and forms a polytomy at node $p(s)$ with three children: the sister taxa of $s$, the left child of $s$, and the right child of $s$.

#### 3.1.1.2. Distinguishing ILS from Gene Events

I represent the species nodes in which ILS could occur with non-binary species nodes. For gene nodes mapped to these species nodes, duplication, transfer and loss are only inferred if the topological disagreement between genes and species cannot be explained completely by ILS.

Conceptually, to distinguish between ILS and other events, I apply a general rule: Given a gene node $g$ and mapped to polytomy species node $\mathcal{M}[g]$, if there exists a binary branching pattern, or resolution, for $\mathcal{M}[g]$ that agrees with the branching pattern of $g$, then ILS is possible and no gene event is inferred.

In practice, generating and testing all binary resolutions of a polytomy would be unwieldy because the number of possible resolutions increases exponentially with the size of the polytomy. Instead, I invoke a test based on the set of descendants of $s = \mathcal{M}[g]$ that have vertically inherited $g$ (Vernot et al., 2008). I call this set $\widehat{N}$. Intuitively, given a gene node $g$ and its two children $l(g)$ and $r(g)$, if a speciation event occurred in $\mathcal{M}[g]$, then each child of $\mathcal{M}[g]$ can only inherit this gene as either $l(g)$ or $r(g)$. If $s = \mathcal{M}[g]$ is a polytomy of size $k$, then each child of $g$ can be retained in a subset of the children of $s$, ranging from size 1 to $k - 1$. These subsets of children species of $s$ that inherit $l(g)$ and $r(g)$ are $\widehat{N}(l(g), s)$ and $\widehat{N}(r(g), s)$, respectively. $\widehat{N}(l(g), s)$ and $\widehat{N}(r(g), s)$ should not overlap if only ILS or speciation has occurred. An overlap between these two sets suggests that the overlapped species inherited both genes $l(g)$ and $r(g)$; therefore, a duplication or transfer event must have occurred. If only duplications, ILS, and speciation were occuring, from the sets of the

two children gene nodes, $\widehat{N}(g,s)$ can be any set that is in the superset of the union of both $\widehat{N}(l(g),s), \widehat{N}(r(g),s)$ and that is also a subset of $C(s)$. Formally, let $C(s)^+$ be all improper subsets of $C(s)$, then all possible values of $\widehat{N}(g,s)$, denoted as $\mathcal{N}(g,s)$, are:

$$\mathcal{N}(g,s) = \{x \subseteq C(s)^+ | x \supseteq \widehat{N}(l(g),s) \cup \widehat{N}(r(g),s)\}. \tag{3.1}$$

This step to determine $\widehat{N}(g,s)$ requires that all $g$, $l(g)$, and $r(g)$ are mapped to the polytomy species node $s$. If $l(g)$ is not mapped to $s$, a climb step is applied to calculate $\widehat{N}(l(g),s)$ from $\widehat{N}(l(g),\mathcal{M}[l(g)])$. Formally, given a gene node $g$ mapped to species node $m = \mathcal{M}[g]$, such that $s$ is an ancestor of $m$, $\widehat{N}(g,s)$ is defined as :

$$\widehat{N}(g,s) = \{x \in C(s) | x >_S m\}. \tag{3.2}$$

Conveniently, this climb process can also infer the number of losses $N_L$ associated with $s$ and $\widehat{N}(g,s)$. If $m$ is a binary species node, then $N_L = h(s) - h(m) - 1$. However, if $m$ is a polytomy, and $\widehat{N}(g,m)$ is not $C(m)$, the full set of all children of $m$, then an extra loss occurred in $m$ that is lost in species $C(m) \setminus \widehat{N}(g,m)$. Therefore the $N_L$ is incremented by 1 to compensate for this difference (see Algorithm 8).

As a new set of rules are introduced in the decision making for gene events, the *generateCostTable* function (see Algorithm 9) needs to be modified for these changes.

- Given a gene $g$, its children $l(g)$ and $r(g)$, and the mappings $\mathcal{M}[l(g)]$ and $\mathcal{M}[r(g)]$, if $\mathcal{M}[l(g)]$ and $\mathcal{M}[r(g)]$ are comparable, it is no longer true that a duplication event must have occurred. An extra check for overlap between $\widehat{N}(l(g),s)$ and $\widehat{N}(r(g),s)$ is necessary to distinguish ILS and duplication (line 17).

- Transfer events need to consider $\widehat{N}$ of the donor and recipient. Given a gene $g$, and its mapping $s$, instead of inferring a transfer from one species to another, the $\widehat{N}(g,s)$, $\widehat{N}(l(g),s)$ and $\widehat{N}(r(g),s)$ are also important to be considered (line 12-15).

These new rules requires that $\mathcal{K}$ and $\mathcal{H}$ are not two-dimensional for each pair of gene node and species node. Instead, for each pair of $g$ and $s$, a third dimension indexed by the value of $\widehat{N}(g,s)$ is also needed (see Algorithm 9 line 30-33) In addition, when calculating the tables $\mathcal{K}$, $\mathcal{H}$ for a gene node $g$, besides the regular information in $\mathcal{H}_g[s]$ used in DTL reconciliation, the $\widehat{N}(r(g),s)$ and $\widehat{N}(l(g),s)$ are also necessary. The tuple that is stored in $\mathcal{H}$ is also expanded to store the $\widehat{N}$ for left and right children gene node.

**Algorithm 8** Climb

```
1 climb(N̂,start,s) {
2     if start = s return N̂
3     N_L = h(s) − h(start)-1
4     if start is non-binary and C(start) ≠ N̂ :   N_L++
5     for each c ∈ C(s) {
6       if start ≤_S c:   return ({c}, N_L)
7     }
8 }
```

**Algorithm 9** DTLI algorithm

```
1 generateCostTable(g, s_1, s_2, Nset_l, Nset_r) {
2     leftCost = K(l(g), s_1), rightCost = K(r(g), s_2)
3     lca = LCA(s_1, s_2)
4     (left, N_L1)=climb(Nset_l, s_1, lca)
5     (right, N_L2)=climb(Nset_r, s_2, lca)
6     Nset=left∪ right
7     N_L = N_L1 + N_L2
8     if s_1 ⋦_S s_2 {
9       # Speciation
10      cost= C_λ N_L +leftCost+rightCost
11      update(g, lca, Nset, s_1, s_2, left, right, cost, "S")
12      # Transfer
13      cost= C_τ
14      update(g, s_1, s_1, s_2, N̂(l(g), s_1), cost, "T")
15      update(g, s_2, s_1, s_2, N̂(r(g), s_2), cost, "T")
16    }else{
17      if left ∩ right ≠ ∅{
18        # Duplication
19        if Nset\right≠ ∅:   N_L++
20        if Nset\left≠ ∅:   N_L++
21        cost= C_δ + C_λ N_L+leftCost+rightCost
22        update(g,lca, Nset, s_1, s_2, left, right, cost, "D")
23      }else{
24        cost= C_λ N_L+leftCost+rightCost
25        update(g,lca, Nset, s_1, s_2, left, right, cost, "S")
26      }
27    }
28 }
29
30 update(g, s, parentSet, s_1, s_2, leftSet, rightSet, cost, ϵ){
31    if cost < K_g[s][parentSet]:   H_g[s][parentSet] = [(s_1, leftSet, s_2, rightSet, ϵ)]
32    if cost = K_g[s][parentSet]:   H_g[s][parentSet].append((s_1, leftSet, s_2, rightSet, ϵ))
33 }
```

### 3.1.2. Traceback

After $\mathcal{K}$ and $\mathcal{H}$ is generated for every gene node, the next step is to generate appropriate $\mathcal{M}$ and $\widehat{N}$ for all gene nodes, while also collecting gene events and total cost. For any gene node $g$, a possible label $s$ and a possible $\widehat{N}$, $nset$, the entry in $\mathcal{K}_g[s][nset]$ represents the lowest cost to map $g$ to $s$ with $nset$. This is equivalent of reconciling a subtree $T_G(g)$ with $T_S$ and $\mathcal{M}[g] = s$. Therefore, the lowest cost solution, (a candidate solution) can be obtained by mapping $\rho_G$ to any species node that yields the lowest cost in $\mathcal{H}_g$.

The detailed algorithm works as follows: First, $M(\rho_G)$ is set to $s*$, such that:

$$s* = \underset{s \in V_S, nset \in C(s)^+}{\operatorname{argmin}} \{\mathcal{K}_{\rho_G}[s][nset]\}. \tag{3.3}$$

Identifying this label and $\widehat{N}$ at $\rho_G$ initiates the traceback procedure, and information is extracted from the entry $\mathcal{H}_{\rho_G}[s][nset]]$, and assigned to the $\mathcal{M}[l(\rho_G)]$, and $\mathcal{M}[r(\rho_G)]$. Recall that a given entry in $\mathcal{H}_{\rho_G}[s][nset]]$ specifies the $\mathcal{M}[l(\rho_G)]$,$\widehat{N}(l(\rho_G), s)$ , $\mathcal{M}[\rho_G]$, and $\widehat{N}(r(\rho_G), s)$ that resulted in an optimal reconciliation of $T_G$ with $T_S$ when $\mathcal{M}[\rho_G] = s$. Therefore, given an entry for gene node $g$, I can determine the $\mathcal{M}$ and $\widehat{N}$ for the left and right children. By recursively repeating this procedure, a reconciled tree $T_R$, augmented with events and mappings to $T_S$, is produced. Once the gene tree is augmented with $\mathcal{M}$ and $\widehat{N}$ for every gene node, the location of gene duplication, transfer, and losses can also be determined in the gene tree. Our algorithm uses the heuristic approach to infer losses, as discussed in Vernot et al. (2008). This heuristic is exact for $\mathcal{M}[g]$ with less than four children; when $\mathcal{M}[g]$ has 4 or more children, it infers the optimal set of losses most of the time.

### 3.1.3. Checking for Temporal Consistency

The validation procedure also consists of two steps: constructing the timing graph and topological sorting. The timing graph $G_T = (V_T, E_T)$ encodes all species nodes in $V_S$ and all temporal relationships between species that are represented by transfers and the species tree. Both constructing and topological sorting have complexity of $O(|V_G| + |V_S|)$. In the worst case, finding one solution that is temporal feasible for a DTL reconciliation problem takes $O(N(|V_G| + |V_S|))$ time, where $N$ represents the total number of candidate solutions. The

validation step under the DTLI model is the same as the validation under the DTL model, as the temporal constraints brought by inferred transfers are not affected by ILS. Here I describe an algorithm for the validation step that I implemented in Stolzer (2011) and also provide pseudocode. This approach is based on a cycle checking scheme proposed by Tofigh (2009) for a less restrictive DT event model. Our approach for DTL reconciliation poses additional constraints because loss events also affect the position of a transfer event. For example, all three scenarios in Figure 3.1 are temporally infeasible under our cycle checking approach, but only (b) violates the constraints described in Tofigh (2009). For example, in Figure 3.1(a), lifting the transfer recipient on edge ($\beta$,C) so that it enters the edge ($\delta$,$\beta$), above the other transfer, would resolve the temporal infeasibility but incur a loss in species D. Under DT event model, this would still be optimal and feasible, but it is not in DTL event model. To generate the timing graph $G_T = (V_T, E_T)$, I first add all species nodes into

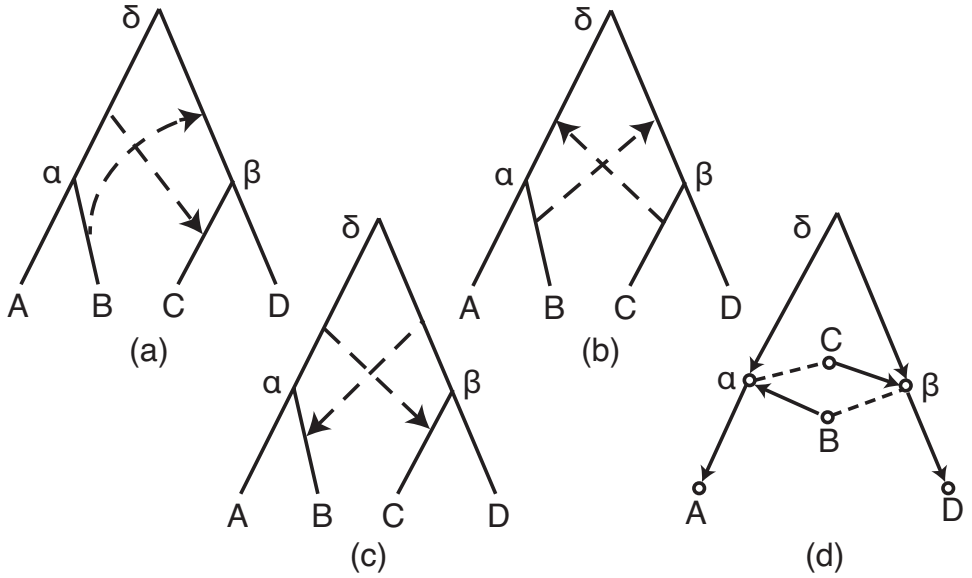



Figure 3.1: (a-c) Temporally infeasible transfer pairs. Here, dashed arrows indicate inferred transfers. (d) Timing graph for the reconciliation in (b). Here, open circles are members of $V_T$, solid arrows are members of $A_T$, and dashed edges are members of $E_T$.

the graph so that $V_T = V_S$. Then I add the timing constraints from the species tree and transfers according to three rules:

1. Timing constraints from the species tree:
   A species node should always pre-date its children. Therefore, for each branch $< s_i, s_j > \in E_S$ in the species tree, add edge $< s_i, s_j >$ to the timing graph $E_T$.

2. Timing constraints from transfers:
   Given a gene tree branch $< p(g), g >$, if $\mathcal{M}[p(g)]$ and $\mathcal{M}[g]$ are incomparable, then it must mean a transfer is inferred from $p(g)$ to $g$. The donor species ($d = \mathcal{M}[p(g)]$) and the recipient species ($r = \mathcal{M}[g]$) must have co-existed. Therefore, any species node that pre-dates $d$, should also pre-date $r$. Two constraints are added: $< p(r), d >$ and $< p(d), r >$.

3. Timing constraints from gene tree:
   Given two transfers on two gene branches $< u, v >$ and $< u', v' >$, if $u$ is an ancestor of $u'$ (formally $u >_G u'$), then $d = \mathcal{M}[u]$ and $r = \mathcal{M}[v]$ must pre-date both $d' = \mathcal{M}[u']$ and $r' = \mathcal{M}[v']$. Four constraints are added: $< p(d), d' >, < p(d), r' >, < p(r), d' >$, and $< p(r), r' >$.

Specifically (Algorithm 10), for every $s \in V_S \setminus L(V_S)$, $< s, l(s) >$ and $< s, r(s) >$ are added to $E_T$ as directed edge. Then all transfer recipient genes are collected by enumerating the event $\mathcal{E}$ table. For transfer recipient gene $g$ and its donor $p(g)$, the mappings are $\mathcal{M}[g]$ and $\mathcal{M}[p(g)]$. The algorithm then adds edges according to rule 2 (line 7). For every pair of transfers, timing constraints are added to $E_T$ according to rule 3 (line 12).

**Algorithm 10** Cycle Checking

```
1  cycleCheck(T_S, T_G):
2      G_T = (V_S, E_S)
3      for each g ∈ V_G \ L(T_G):
4          if E[g] = T:
5              if M[l(g)] ⋠_G M[g]:  r = M[l(g)]  else:  r = M[r(g)]
6              d = M[g]
7              add edge < p(r), d > and < p(d), r > to E_T
8              for each g' ∈ V_G \ L(T_G)  s.t.  g ≥_G g':
9                  if E[g'] = T:
10                     if M[l(g')] ⋠_G M[g']:  r' = M[l(g')]  else:  r' = M[r(g')]
11                     d' = M[g']
12                     add edge < p(d), d' >, < p(d), r' >, < p(r), d' >, < p(r), r' > to E_T
13     order=topologicalSort(G_T)
14     if order exist:  return true; else:  return false;
```

## 3.2. Speeding Up Cycle Checking

While good accuracy and complexity are the two major factors to consider when developing an algorithm, the implementation of cycle checking functionality in software requires more considerations, such as language, memory limitations, user interface, and software design. Here I describe two techniques that did not change the complexity of the algorithm, but significantly reduced runtime in the cycle checking process.

### 3.2.1. Constructing Timing Graph: Saving Species Tree

In the process of building the timing graph for one solution, a total of $|E_S| + 2N_T + 4{N_T}^2$ timing constraints are added. Considering multiple candidate solutions, especially a batch of reconciliations where many gene trees are reconciled with the same species tree, this process is repetitive and time consuming. The $|E_S|$ edges are always added for each cycle checking process. Therefore, instead of creating a new timing graph for each cycle checking process, reusing the $|E_S|$ branches in the same timing graph could be a practical way to save the calculation overhead. However, reusing these branches would require a clean up step after each cycle checking, where the previous transfer timing constraints are removed. This creates a dilemma of two options:

- Option 1: Memoize timing constraints from the species tree in $E_T$, then add timing constraints introduced by transfers for cycle checking. At the end of the process, remove transfer timing constraints involved in this solution from $E_T$.

- Option 2: Do not memoize timing constraints. For every cycle checking process, initialize a new timing graph and add timing constraints from the three rules.

In the case where $N_T$ is greater than $|V_S|$, option 1 would take more time to finish because of the extra steps of removing transfer timing constraints; but in the case where $N_T$ is much smaller than $|V_S|$, option 1, memoizing species tree branches, would save time in practice. According to a timing analysis using 4000 cyanobacterial gene trees and the same species tree with 64 species, in practice, memoizing the timing constraints from the species tree is a good option to speed up cycle checking in large scale reconciliation.

### 3.2.2. Cycle Checking: Stop Before Checking the Entire Timing Graph

The goal of this process is only to verify whether the timing graph is acyclic. Because a single temporal conflict invalidates the solution, the process can exit as soon as one conflict is found. The complexity of the existing topological sorting algorithm is linear with the number of arcs in the timing graph, in $O(|V_T| + |E_T|)$ (Cormen et al., 1990). That algorithm generates an ordered list of all species nodes that satisfy all temporal constraints. However, small cycles, such as temporal conflicts between two transfers, can be easily detected during the construction of the timing graph. When adding the third constraint, if any pair of transfers exhibit a small cycle, (for example, if the child of an transfer is the ancestor of the other and vice versa), then the cycle checking is immediately terminated. In a batch reconciliation of 4000 gene trees, applying these two modifications of cycle checking significantly improves the speed of the validation step, reducing the run-time by almost 5 fold.

## 3.3. Domain Tree Level Reconciliation

In addition to genes and species, a third level of evolution exists in genomes: domain evolution. A domain is a segment of sequence that acts as a modular unit in a gene, folding independently and performing a specific function. A single gene may have one or more domains; in the latter case, the gene is then referred to as a multi-domain gene. The content of domains in a multi-domain gene is referred to as its domain architecture. At the domain level, evolution occurs via domain shuffling events: domain insertion, transfer, duplication, and deletion, as well as gene fusion and fission (Gilbert, 1978, 1987; Kaessmann et al., 2002; Long et al., 2003; Patthy, 2003; Stolzer et al., 2015; Thornton and DeSalle, 2000; Tordai et al., 2005; van Rijk and Bloemendal, 2003). Molecular processes underlying these abstract events include non-allelic homologous recombination, retro-transposition, read-through errors, and unequal crossing-over. Understanding how domain architectures in multi-domain gene families evolve via domain shuffling, in addition to events on the gene and species levels, is crucial to understanding the evolution of multi-domain family proteins.

For understanding domain shuffling events, I define a general problem: Given rooted domain, gene, and species trees, an event model, and tables that indicate the presence and location of domains in genes and the presence of genes in species, the goal is to find the event histories of both the domain and gene families that best explain the topological difference between

the domain tree, gene tree, and species tree.

In a phylogenetic context, domain shuffling events appear very similar to the gene events that occurred in species discussed in Chapter 2. For example, the domain insertion represents a domain from one gene being copied and inserted to a different gene, which is analogous to a gene transfer from one species to a different species. These similarities suggests a phylogenetic reconciliation algorithm can be applied to the domain context and infer domain shuffling events. However, this level of evolution is different from gene tree reconciliation in several aspects. First, some domain events, such as horizontal acquisition of a domain, may not only occur between genes, but between species, as well. This increases the hardness of the problem, as the reconciliation between the domain tree and the gene tree will also require species tree information to distinguish these events. Second, a given domain family may occur in only one gene family, or it may occur in multiple different gene families. If a domain exists in only one gene family, this problem can be treated similarly to gene tree reconciliation with species tree. However, if a domain family exists in multiple gene families, the underlying evolutionary relationship between these gene families is hard to estimate. The relationship of genes in the same gene family can be constructed by tree building methods from sequence alignment information, whereas the relationship of genes in different gene families has more obstacles, such as different function, different sequence composition, and different sequence length. This complicates the problem of inferring domain shuffling events as it is unclear how to define a reference tree. Here, I focus on a more specific problem that is restricted to domain families that occur in single gene families, or domain families that occur across multiple gene families, but the copies in each gene family form a clade in the domain tree, minimizing the interaction of domain shuffling events across different gene families. I also assumes that 1) domain shuffling occurs within existing genes and 2) the gene tree and species tree are correctly estimated. This defines a reconciliation problem on three levels of evolutionary phylogeny.

This general problem of inferring domain shuffling events across domain tree, gene tree, and species tree is hard. More specifically, the optimal gene tree-species tree reconciliation may not result in an optimal domain event history. This suggests a global optimization of both a reconciled gene tree and a reconciled domain tree, which is much harder than standard reconciliation. I provide a step toward solving this general problem, by solving a simpler problem, assuming that the gene tree is correctly reconciled: Given a rooted, binary species tree, $T_S$, a rooted, binary gene tree $T_G^*$ that has been reconciled with $T_S$, a rooted domain

tree, $T_D$, and a mapping $\mathcal{M}_L : L(T_D) \rightarrow L(T_G)$, from the leaves of domain tree to the leaves of the gene tree, find the set of domain shuffling events that best explain the difference between $T_D$ and $T_G^*$.

Here, I describe how the standard DTL $T_G$-$T_S$ reconciliation algorithm can be modified and applied to $T_D$-$T_G$ reconciliation.

### 3.3.1. Model

I first define a model of events underlying the domain tree reconciliation problem:

- Co-divergence ($\Delta$): A bifurcation in the domain tree that arose through a bifurcation in the gene tree, analogous to speciation in $T_G$-$T_S$ reconciliation. The gene tree bifurcation may have arisen via speciation ($\Delta_S$) in the species tree, gene duplication ($\Delta_D$), or gene transfer ($\Delta_T$).
- Duplication (D): A single domain is copied inside a gene resulting in two separate copies of the domain within the same gene, analogous to gene duplication in gene tree reconciliation.
- Deletion (L): A domain is deleted from the gene (and genome), analogous to gene loss in gene tree reconciliation. Note that this event should not be invoked if the loss is due to a gene loss.
- Domain insertion (I): A new copy of the domain is inserted into a different gene within the same genome, analogous to gene transfer.
- Horizontal domain transfer (T): A new copy of a domain is inserted into a gene in a different genome, analogous to gene transfer.

I leave the modeling of gene fusion and fission to future work as they involve a model where one entity has two different parents, and may require additional information, such as genome context. Our model follows the dynamic programming strategy used in other DTL reconciliation problems with modified rules for calculating the cost tables $\mathcal{K}$ and $\mathcal{H}$. In this section, the previously defined $\mathcal{K}$, $\mathcal{H}$, $\mathcal{M}$, and $\mathcal{E}$ tables (Section 2.3.2) that indicate the relationship between genes and species are also extended to indicate the relationship between domains and genes. I use subscript $D$ and $G$ to represent tables associated with

the domain tree and gene tree, respectively. For example, $\mathcal{H}_D[d]$ is a table associated with node $d \in V_D$, and stores the information of mapping gene nodes onto this domain node, $d$.

**Domain Event Inference with Transfers (DE-DTL)**

**Input**:A rooted, binary domain tree, $T_D = (V_D, E_D)$; a rooted, binary, DTL-reconciled gene tree, $T_G^*$ ; and a leaf mapping, $\mathcal{M}$:$L(T_D) \to L(T_G^*)$.

**Output**:The set of all temporally feasible, domain shuffling histories in $T_D$ that minimize

$$C(N_D, N_T, N_L, N_\Delta) = N_D C_D + N_T C_T + N_L C_L + N_\Delta C_\Delta,$$

where $N_\epsilon$ and $C_\epsilon$ represent the number and cost of event type $\epsilon$. To solve this problem, four major challenges must be considered. First, a co-divergence event can be subcategorized as $\Delta_S, \Delta_D$ or $\Delta_T$. This requires additional information about the gene tree bifurcation. Second, domain insertions and domain transfers both occur between two gene nodes that are incomparable to each other; however, a domain insertion occurs between genes in the same genome, whereas a domain transfer occurs between two genes that are in different species. To make this distinction, additional tests based on information from the reconciled gene tree are required to infer the correct events. Third, a missing taxon in the domain tree may be caused by a domain deletion event or the loss of the gene. If the gene is already lost, the domain deletion event should not contribute to the event cost equation for the domain-gene tree reconciliation. Finally, when testing for temporal feasibility, both species tree temporal constraints and gene tree temporal constraints must be considered. For species constraints, in addition to gene transfers and species tree temporal constraints, extra constraints introduced by domain transfers must be considered. For gene tree temporal constraints, the temporal constraints from domain insertion and domain transfers must be considered. Therefore, for a domain level reconciliation, two timing graphs are tested.

### 3.3.2. Algorithm

Here, I introduce an algorithm for the domain event inference problem that also solves the four challenges discussed above in a four step algorithm (Algorithm 11). Note the $\gamma_i$ in this context represents the reconciliation solution of domain-gene tree reconciliation, instead

of previously defined result of gene-species tree reconciliation. I now describe each step in detail.

#### 3.3.2.1. Construct Modified Gene Tree: $T_G^*$

After the gene tree has been reconciled, the $\mathcal{E}_G$ and $\mathcal{M}_G$ tables effectively allow the domain tree reconciliation to distinguish between the different types of co-divergence events. The species where each gene resides and the events that occurred on each gene node can be found by a simple look up in those two tables. These two tables can also be used to determine whether or not the donor gene and recipient gene of a horizontal domain event are in the same genome. This is accomplished by looking up the species label on the donor and recipient gene.

The function *addLoss*(line 1) modifies the reconciled gene tree $T_G$ and adds extra gene leaves and internal gene nodes to the gene tree to represent the taxa that are missing due to gene losses. New leaves, referred to as pseudoleaves, represent the lost gene(s), while new internal nodes, referred to as pseudonodes, represent ancestral genes that are "missing" due to gene loss. The result of the modification generates is a modified gene tree, denoted $T_G^*$. These added gene losses are used to distinguish between domain loss and gene loss. If a domain that is lost is mapped to a pseudoleaf, then this domain is lost due to gene loss and therefore not counted in domain loss.

**Algorithm 11** Domain Tree Reconciliation

```
DE-DTL(T_D, T_G, T_S) {
 1    T_G^* = addLoss(T_G, T_S)
 2    generateCostTable(T_D, T_G^*)
 3    {γ_1, γ_2...γ_m} = traceback(K, T_D, T_G)
 4    {γ_1, γ_2...γ_n} = checkFeasibility({γ_1, γ_2...γ_m})
 5 }
```

The addition of pseudonodes also allows for precise estimation of the association between a gene node and domain node. For example, consider the species tree, modified gene tree, and domain tree shown in Figure 3.2. I can infer that the gene node associated with node $u$ in $T_D$ was on the gene lineage between $g1\_E$ and $\phi R$. This gene branch is in the species lineage between Eudicots and Rosids. In this inference, I can estimate the time that the domain insertion occurred on $u$ to be between Eudicots and Rosids in species tree. Without the

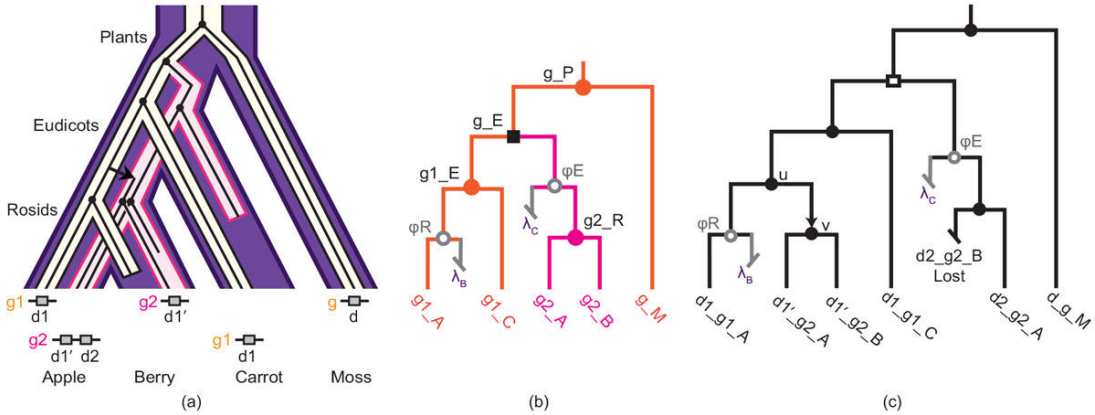



Figure 3.2: An example of domain insertion. (a) Species tree with embedded gene and domain trees showing the co-evolution of domains, genes, and species in a hypothetical family. Present day genes in the family, including their domain content, are shown on the leaves of the species tree. (b) The modified gene tree $T_G^*$. The pseudonodes are $\phi R$ and $\phi E$; the two pseudoleaves are $\lambda_B$ and $\lambda_C$. They represent the location of the missing taxa in the gene tree. Gene duplication occurred on $g\_E$ and is represented as a black square; filled circles represent co-speciation between gene and species. (c) A reconciled domain tree, showing a domain insertion on edge $< u, v >$ and three domain losses ($\lambda_B$, $\lambda_C$, and $d2\_g2\_B$). The $\lambda_B$ and $\lambda_C$ are domains that are missing due to gene loss in (b), therefore, they are not counted when calculating the reconciliation cost. Co-divergence due to gene duplication is represented by an open square.

pseudonode $\phi R$, the domain node $u$ would be on the gene lineage between $g1\_E$ and $g1\_A$, which is on the species lineage between *Eudicots* and *Apple*. Then it would not be possible to determine whether or not $u$ predates Rosids, the divergence between Apple and Berry. I now describe the *addLoss* function in detail.

**AddLoss**: The $T_G^*$ is constructed by traversing the gene tree $T_G$ and visiting every gene node. For every gene node $g$ and its parent $p(g)$ that are not related through a horizontal gene transfer event, I can calculate the loss nodes by analysing the species mapping on $g$ and $p(g)$. If speciation occurred on $p(g)$, then $\mathcal{M}[p(g)]$ should be the parent of $\mathcal{M}[g]$ if no loss occurred. Otherwise, the extra height difference is caused by loss of genes. For each loss node, I add an ancestral gene $\phi$ in the branch that represents the gene that was present in the species lineage from $\mathcal{M}[p(g)]$ and $\mathcal{M}[g]$, but is not observable due to gene loss. For each pseudo ancestral gene that is mapped to a species node $s \in V_S$, I also add a pseudoleaf gene $\lambda_{\hat{s}}$, where $\hat{s}$ is the left child of $s$, if $\mathcal{M}[g] \not\leqslant l(s)$, otherwise $\hat{s}$ is the right child of $s$. Note that this pseudoleaf that is mapped to $\hat{s}$ may correspond to an internal node on $T_S$ to represent a loss of entire clade of species.

#### 3.3.2.2. Generating the Cost Tables $\mathcal{K}_D$ and $\mathcal{H}_D$

The function **generateCostTable** (line 2) generates the cost tables $\mathcal{K}_D$ and $\mathcal{H}_D$ for the domain tree. This process also follows the dynamic programming framework and traverses the domain tree in post-order, just like regular DTL reconciliation for a gene tree. However, the rules for domain shuffling events are slightly different from the rules for gene events, discussed in Section 2.3.2.1. Given a domain $d$ and its two children $d_1$ and $d_2$, and their gene nodes $g$, $g_1$ and $g_2$, which are mapped to $s = \mathcal{M}_G[g]$, $s_1 = \mathcal{M}_G[g_1]$, and $s_2 = \mathcal{M}_G[g_2]$, the possible events on $d$ are defined as follows:

- If $g_1 \not\leq_G g_2$ and $\text{LCA}(g_1, g_2) >_G g$, then either a domain transfer or domain insertion has occurred on $d$. If $g \geq_G g_2$, then the domain insertion (or transfer) is inferred between $d$ and $d_1$, and domain losses (if any) are inferred between $d$ and $d_2$. If $\mathcal{M}_G[g]$ and $\mathcal{M}_G[g_1]$ are the same species, then a domain insertion is inferred on $d$, otherwise a domain transfer is inferred. The inverse is true for the case where $g \geq_G g_1$.

- If $g_1 \not\leq_G g_2$ and $g \geq_G \text{LCA}(g_1, g_2)$, then a co-speciation has occurred on $d$. The further sub-category of co-speciation $(\Delta_S, \Delta_D, \Delta_T)$ can be assigned by a simple lookup in $\mathcal{E}[g]$. Domain losses (if any) are inferred between $g$ and $\text{LCA}(g_1, g_2)$.

- If $g_1 \geq_G g_2$ or $g_2 \geq_G g_1$, then a domain duplication is inferred and $g$ must be the ancestor of both $g_1$ and $g_2$ ($g \geq_G \text{LCA}(g_1, g_2)$). Domain losses (if any) are inferred in three branches, between $g$ and $\text{LCA}(g_1, g_2)$, between $g$ and $g_1$, and between $g$ and $g_2$.

Note that when domain losses are inferred on a domain branch $< p(d), d >$, it is equivalent to inferring domain losses between two gene nodes $\mathcal{M}_D[p(d)]$ and $\mathcal{M}_D[d]$. The number of domain losses inferred between these two gene nodes is the number of non-pseudonode between them.

#### 3.3.2.3. Traceback to Infer Domain Events

The traceback algorithm in line 3 follows the same backtracking framework that DTL-reconciliation uses and that is discussed in Section 3.1.2. The domain tree $T_D$ is traversed top to bottom, and a candidate reconciliation is generated from $\mathcal{H}_D$, populating entries in $\mathcal{M}_D$ and $\mathcal{E}_D$. The slight modification is the extra events that are assigned to the event table for each domain node. To infer the domain losses on a domain branch $< p(d), d >$ for each

ancestral node $g$ that is missing between $\mathcal{M}_D[d]$ and $\mathcal{M}_D[p(d)]$, a domain loss is inferred in $g'$, the child of $g$ that is incomparable to $\mathcal{M}_D[d]$. If $g$ is a pseudonode in $T_G^*$ or $g'$ is a pseudoleaf in $T_G^*$, then $g'$ is a gene loss. Otherwise, a domain loss in $g'$ is inferred.

#### 3.3.2.4. Checking Temporal Feasibility

The three-level reconciliation algorithm introduces extra temporal constraints for domains, genes, and species. For a reconciliation of the domain tree to be valid, it must satisfy two criteria. First, it must be possible to order the species nodes on a time line such that the temporal constraints imposed by gene transfers and domain transfers are consistent with the time line. Second, it must be possible to order the gene nodes in a time line, such that the temporal constraints imposed by domain transfers and domain insertions are consistent with the time line.

For species temporal consistency, the species timing graph (Section 2.3.2.3) is modified by adding extra constraints that are introduced by domain transfer events in addition to those imposed by the gene tree reconciliation. Then the topological sorting algorithm described in Section 2.3.2.3 can be applied to the timing graph and checked for temporal feasibility.

For gene temporal consistency, I introduce a gene timing graph $G_T^G = (V_T^G, E_T^G)$ that is constructed from the timing constraints implied by the gene tree, the domain insertion events, and the domain transfer events. The timing graph edges are added to represent the following three temporal constraints:

- For every gene node $g$ in the modified gene tree $T_G^*$ (which includes pseudonodes) the parent, $p(g)$, must predate $g$. Therefore I add $< g, p(g) >$ to $E_T^G$ $\forall g \in V_G \setminus \{\rho\}$.
- If a domain was transferred or inserted from gene $g_i$ to $g_j$, then $g_i$ and $g_j$ must have co-existed. This condition is satisfied, if the parent of $g_i$ pre-dates $g_j$, and parent of $g_j$ pre-dates $g_i$. Therefore, for each $< g_i, g_j >$ that is an inferred domain transfer or insertion event, I add $< p(g_i), g_j >$ and $< p(g_j), g_i >$ to $E_T^G$.
- Let $< d_1, d_2 >$ and $< \hat{d}_1, \hat{d}_2 >$ be two horizontal domain events (i.e., insertions or transfers) in the domain tree such that $d_2 \geq_D \hat{d}_1$. Then, the two genes involved in the first domain event $< d_1, d_2 >$, should pre-date the two genes involved in the second domain event $< \hat{d}_1, \hat{d}_2 >$. Therefore, I add $< p(g_1), \hat{g}_1 >, < p(g_1), \hat{g}_2 >, < p(g_2), \hat{g}_1 >, < p(g_2), \hat{g}_2 >$ to $E_T^G$.

Once $G_T^G$ is constructed, the temporal feasibility checking (discussed in Section 3.1.3) is used to check for cycles in the gene timing graph. If any cycles exist in either the species or the gene timing graph, the candidate history is temporally infeasible and not reported.

# Chapter 4

# Rearrangement Algorithm to Fix Weakly Supported Branches

When both the gene tree and species tree are correct, incongruence reflects the difference between the histories of the genes and species. However, incongruence can also arise through phylogenetic error in the gene tree due to insufficient or incorrect data or systematic bias. In this case, a gene tree branch may disagree with the species tree. When using such a tree for reconciliation, gene events may be incorrectly inferred.

In this chapter, I present my novel work of correcting the incongruence in gene trees that may be caused by uncertainty. These results has been accepted for publication (Lai et al., 2017). To place this work in context, I first review ways to assess uncertainty in gene trees and briefly discuss prior methods that consider uncertainty in gene trees when performing reconciliation under the DL and DTL event models. Then, I describe an exact algorithm and heuristic methods that I developed for resolving uncertainty in gene trees under DTL event model. I also analysed the performance of heuristic strategies compared to the optimal, exhaustive strategy. The performance and runtime of these methods are compared using two data sets: a phylogenomic cyanobacteria data set (Latysheva et al., 2012) and the simulated data set used in Szöllõsi et al. (2013a).

## 4.1. Representing Uncertainty in Gene Trees

There are many ways to represent uncertainty in a gene tree, which generally fall into two categories: non-binary nodes and low measures of branch support. In the former case, a non-binary node is a soft polytomy and is the result of insufficient evidence for the branching order. For an in-depth discussion on soft versus hard polytomies, refer to Section 2.1.1.1. Since a non-binary node does not indicate which lineage diverged first, all binary resolutions of the same size may be appropriate. A gene tree with soft polytomies is also called an unresolved gene tree.

Another way to represent uncertainty in a gene tree is with branch support. Weak branch support suggests that the sequence data does not contain a strong enough signal to determine the branching order (Anisimova et al., 2011). Common methods to assess branch support include the bootstrap and likelihood scores. See Anisimova et al. (2011) for a review. This representation of uncertainty also suggests that many possible branching orders may be appropriate alternative hypothesis for uncertain branch(es). This representation can be easily converted to a non-binary tree by collapsing the weakly supported branches (i.e., branches with support below a pre-specified threshold) (Figure 4.1).

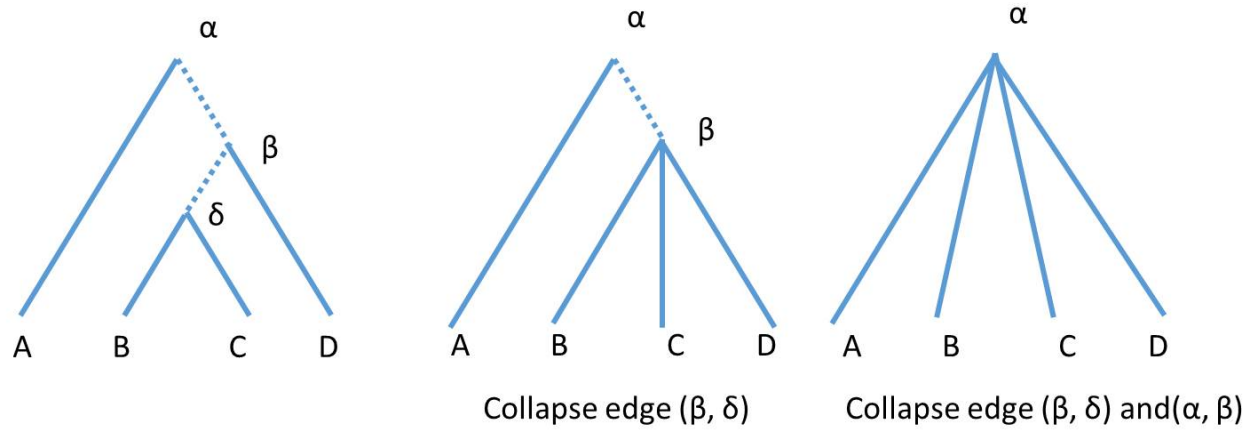



Figure 4.1: A binary gene tree with branches with two low support (dashed lines) can be collapsed and result in a non-binary gene tree with a polytomy of size 4.

## 4.2. Background on Resolving Uncertainty

If the molecular data does not contain enough information to resolve the branching order, perhaps other information can be used to generate a hypothesis for the true branching order. Given a non-binary gene tree, what is the best binary resolution of the tree?

Different binary resolutions from a non-binary gene node may result in different conclusions when used in other downstream evolutionary analyses. Therefore, it is important to generate the binary resolution that makes the most sense.

For example, when reconciling the gene tree with the species tree, different resolutions will result in different inferred event histories, and some may be more parsimonious than others. Thus, reconciliation provides one principled framework for choosing a binary resolution of a non-binary gene tree or gene tree with weakly supported branches: under a parsimony criterion, the resolved binary gene tree with the binary resolution that minimizes the cost of event inference is the best gene tree (Chen et al., 2000).

This conservative approach effectively avoids the scenario that a binary resolution may disagree with the species tree, leading to inferred gene events, when the incongruence could be due to insufficient data.

Species-tree aware methods seek to improve the accuracy of gene tree inference by exploiting species tree information, without erasing evidence of the biological processes that contributed to incongruence. This approach is based on the assumption that in the absence of strong evidence that supports an alternative branching order, I expect the gene tree should match the species tree. Species-tree aware methods embody different trade-offs between speed and accuracy, ranging from detailed probabilistic models (Boussau et al., 2013; Gorecki and Eulenstein, 2012; Rasmussen and Kellis, 2011; Sjöstrand et al., 2012; Sjőstrand et al., 2014; Szöllõsi et al., 2013a) to fast approaches based on parsimony or other criteria (Chaudhary et al., 2012; Jacox et al., 2017; Kordi and Bansal, 2016; Lafond et al., 2016; M. et al., 2012; Noutahi et al., 2016; Scornavacca et al., 2015; Swenson et al., 2012; Thomas, 2010; Vilella et al., 2009; Wapinski et al., 2007; Zheng and Zhang, 2014a).

Here I focus on parsimony approaches designed to correct an unresolved gene tree. Corrective strategies are applied in two steps: First, a gene tree is constructed from sequence alignments using molecular phylogenetic methods. In a second, corrective step, species tree information

is used to improve the quality of the gene tree. By separating consideration of sequence evolution and gene events, and focusing only on those areas where the sequence data cannot resolve the topology, the search space is substantially reduced. Formally, the cost of a reconciliation between a gene tree and species tree is defined to be:

$$\kappa = C_D \cdot N_D + C_T \cdot N_T + C_L \cdot N_L, \tag{4.1}$$

where $N_\epsilon$ and $C_\epsilon$ are the number and cost of event type $\epsilon$. Under the DL event model, the $C_T \cdot N_T$ term is ignored because transfer is not considered.

Let $T_G = (V_G, E_G)$ be a non-binary, rooted gene tree. A binary, rooted gene tree, $T_G^b = (V_G^b, E_G^b)$, is a *binary resolution* of $T_G$, if for every binary node $g \in V_G$, there exists a node $g' \in V_G^b$, such that $L(T_G^b(l(g'))) = L(T_G(l(g)))$ and $L(T_G^b(r(g'))) = L(T_G(r(g)))$. The set of all possible binary resolutions is denoted $\mathfrak{T}(T_G)$. For a polytomy, $g$, $\mathfrak{T}(g)$ denotes the set of all binary trees with $k(g)$ leaves, and the leaf set is connected to $C(g)$, the children set of $g$. The problem of finding the resolution of a gene tree with the lowest cost reconciliation under the DL and DTL event models are, respectively, as follows:

**Problem statement: Resolve-DL**

**Input:** An unresolved gene tree $T_G = (V_G, E_G)$, a binary species tree $T_S$, a mapping $\mathcal{M}_L : L(T_G) \to L(T_S)$; and event costs $C_\epsilon$ for each event $\epsilon \in \{D, L\}$.

**Output:** $\mathfrak{T}^*(T_G)$=$\{T_G^*\}$, the set of all binary resolutions of $T_G$, such that $T_G^* \in \mathfrak{T}(T_G)$,

$$T_G^* = argmin_b\ \kappa(T_G^b, T_S),$$

under DL event model.

**Problem statement: Resolve-DTL**

**Input:** An unresolved gene tree $T_G = (V_G, E_G)$, a binary species tree $T_S$, a mapping $\mathcal{M}_L : L(T_G) \to L(T_S)$; and event costs $C_\epsilon$ for each event $\epsilon \in \{D, T, L\}$.

**Output:** $\mathfrak{T}^*(T_G)$=$\{T_G^*\}$, the set of all binary resolutions of $T_G$, such that $T_G^* \in \mathfrak{T}(T_G)$,

$$T_G^* = argmin_b\ \kappa(T_G^b, T_S),$$

and the inferred events are temporally feasible.

For both problems, the aims are to resolve the polytomies by replacing them with binary

resolutions to minimize the reconciliation cost. The number of possible binary resolutions for a polytomy increases exponentially with the size of the polytomy. The resolve-DL problem was first proposed by Chen et al. (2000). The first exact algorithm were developed by Durand et al. (2005) and showed that the resolve-DL problem can be solved in polynomial time (Durand et al., 2006a). Several algorithmic advances followed for this problem (Chaudhary et al., 2012; Lafond et al., 2016; M. et al., 2012; Noutahi et al., 2016; Swenson et al., 2012; Zheng and Zhang, 2014a).

However, the Resolve-DTL problem is much more complicated. Under the DL event model, the ancestor of a gene must reside in an ancestor of the species in which the gene resides. Given a gene node $g$, if $\mathcal{M}[g] = s$, then the parent of $g$, $(p(g))$, must be mapped to species $s$ or to an ancestor of $s$. If a transfer event occured in the gene family, this assumption is no longer true. Given a gene node $g$, the parent $p(g)$ may reside in species that is not the ancestor of the species where $g$ resides. This special feature of transfers significantly complicates the search for the best binary resolution. In fact, this problem has been shown to be NP-hard (Kordi and Bansal, 2017). This suggests that to obtain the optimal resolved gene tree, enumerating an exponential number of resolutions is required. Exhaustive search algorithms that are fixed-parameter tractable were introduced for both dated and undated species trees (Jacox et al., 2017; Kordi and Bansal, 2016; Scornavacca et al., 2015). Kordi and Bansal (2016) introduced an exact algorithm that enumerates all binary resolutions of an unresolved gene tree and generates one resolved gene tree that yields the minimum cost. The algorithm guarantees optimal binary gene tree with exponential complexity with polytomy size ($O(2^{k(log_2 2k)}l|V_S|)$), where $k$ and $l$ represent the size of the largest polytomy and number of polytomies, respectively. Jacox et al. (2016) later developed a faster exact algorithm for the Resolve-DTL problem and implemented the algorithm in ecceTERA. Although still exponential, the algorithm significantly reduced the complexity compared to the previous algorithm from Kordi and Bansal (2016) by enumerating a set of tripartitions for unresolved polytomy. This potentially reduces the number of resolutions that are must be enumerated. The complexity of the algorithm is $O(|V_G||V_S|^2(3^k - 2^{k+1})$ and the algorithm is reported to be reasonably fast at resolving gene trees with polytomies of size 12 or less. The focus of the algorithm was mainly for dated species tree, and can be modified to apply to undated species tree.

Several heuristic approaches have also been developed because the exhaustive search gets increasingly expensive as the size of the polytomy increases (Bansal et al., 2015; David and

Alm, 2011; Nguyen et al., 2013). These heuristics can potentially be useful for gene trees with larger polytomies in real data sets.

In TreeFix-DTL, Bansal et al. (2015) implemented a Nearest Neighbour Interchange (NNI) heuristic to search for the best resolution for a polytomy. The algorithm generates candidate topologies; a candidate is accepted if it has a lower reconciliation cost and is statistically equivalent to the most likely tree, based on the sequence data. However, the Resolve-DTL heuristic in TreeFix-DTL is embedded in the framework of gene tree reconstruction as it requires the sequences and the ML gene tree to decide whether the new topology is accepted. Because of this embedding, this method may not be suitable for solving Resolve-DTL problem on gene trees reconstructed using other methods.

Mowgli-NNI resolves each polytomy in the gene tree using a modified NNI method to sample a subset of binary resolutions (Nguyen et al., 2013). The heuristic algorithm uses a hill-climbing-like approach to search for a resolved gene tree and iterates all weak edges in an unresolved gene tree. Each candidate resolution is tested for the best parsimony score. However, Mowgli-NNI requires the species tree to be dated, and correct speciation times are frequently unknown.

In the set of resolutions, there may be multiple binary resolutions that result in the lowest reconciliation cost. Moreover, for each binary resolution, there may be multiple event histories that minimize the reconciliation cost. The problem of degeneracy has been approached from various perspectives. The above algorithms adapted the simplest approach, which is to output a single resolution, selected arbitrarily. However, generating all solutions enables the user to choose the best resolution based on problem-specific criteria. For example, the NOTUNG GUI provides a point-and-click interface that allows the users to investigate alternate resolutions interactively (http://www.cs.cmu.edu/ durand/Notung/). This is a satisfying solution for exploratory analysis of a few families, but unmanageable for large phylogenomic analyses. A third approach is to triage alternate binary resolutions by incorporating additional information, such as phylogenetic likelihood (Bansal et al., 2015), conditional clade probabilities (Jacox et al., 2017), bootstrap replicates (David and Alm, 2011) or genome neighbourhoods (Chauve et al., 2013; Lafond et al., 2013).

Here, I focus on the problem of resolving a non-binary tree by minimizing inferred events in the DTL event model. Many existing rearrangement algorithms for the DTL event model either require a dated species tree, or given an undated species tree, fail to check for tem-

poral consistency. If a binary resolution with minimum reconciliation cost is not temporally feasible, then it should not be considered a valid solution. The methods presented and implemented here use undated species trees and ensure that the generated solutions to be temporally feasible.

I present new heuristics to correct incongruence in gene trees that may be caused by uncertainty. These novel heuristics, as well as the exact algorithm, have been implemented in NOTUNG. Our exact algorithm generates all temporally feasible, binary resolutions and event histories.

## 4.3. An Exact Algorithm to Solve Resolve-DTL

The algorithm to solve Resolve-DTL (Algorithm 12) consists of the same three steps as the DTL-Reconcile algorithm (described in detail in Stolzer et al. (2012a)). The key modification for Resolve-DTL is a procedure to calculate the cost tables when $g$ is a polytomy. In the first step, all nodes in $T_G$ are visited in post-order, and at each node $g$, all possible assignments of $\mathcal{M}[g]$ and $\mathcal{E}[g]$ are enumerated. The associated information is stored in two tables: $\mathcal{K}_g$, which stores the best cost for mapping $g$ to each $s \in T_S$, and $\mathcal{H}_g$, which stores the corresponding book-keeping information necessary for calculating an event history in step two. If $g$ is a binary gene node, the core subroutine that generates $\mathcal{K}_g$ and $\mathcal{H}_g$ is a function `costCalc`$(g, s_l, s_r)$ (line 9 in Algorithm 12, see also Stolzer et al. (2012a)). The cost of mapping gene node $g$ to species node $s$ only depends on the event at $g$ (i.e. $\epsilon = \mathcal{E}[g]$), the mapping on children of $g$ (i.e. $s_r = \mathcal{M}[l(g)]$ and $s_l = \mathcal{M}[r(g)]$), and the cost associated with these mappings $(\mathcal{K}_{l(g)}[s_l], \mathcal{K}_{r(g)}[s_r])$. This information is stored in $\mathcal{H}_g[s]$ in the form of a tuple: $(\epsilon, s_r, s_l)$.

If $g$ is a polytomy, all possible binary resolutions in $\mathfrak{T}(g)$ are exhaustively enumerated, and $\mathcal{K}_g$ and $\mathcal{H}_g$ are calculated for each $T_g^b \in \mathfrak{T}(g)$. In each binary resolution, $g$ and its children are replaced with a binary embedded subtree with $k_g$ leaves, inserting $k(g)-1$ binary internal nodes (including the root that replaces $g$). Cost tables for these internal nodes are constructed, bottom up, from the cost tables at the leaves of the embedded subtree, which have already been constructed during the post-order traversal, prior to reaching $g$.

When this procedure completes, the root of each $T_g^b \in \mathfrak{T}(g)$, will be annotated with tables $\mathcal{K}_g^b$ and $\mathcal{H}_g^b$. These tables are then combined (line 13 in Algorithm 12) to generate two integrated

cost tables, $\mathcal{K}_g$ and $\mathcal{H}_g$, for polytomy node $g$, as illustrated in Fig. 5.1. For each value of $s$, $\mathcal{K}_g[s] = \min_b \mathcal{K}_g^b[s]$. For those binary resolutions that minimize $\mathcal{K}_g^b[s]$, the information in $\mathcal{H}_g^b[s]$ is added to the combined table $\mathcal{H}_g[s]$. In contrast to binary nodes, for polytomies, the tuples stored in $\mathcal{H}_g[s]$ include the binary resolution(s) associated with the optimal event cost: $(\epsilon, s_l, s_r, T_G^b)$. This extra information is used in the pre-order traversal in the second pass to reconstruct the optimal binary resolution of the gene tree as a whole. When there are multiple entries that result in the same cost for a pair of $g$ and $s$ in $\mathcal{K}_g[s]$, these tuples are all stored in $\mathcal{H}_g[s]$ as a list (for both binary nodes and polytomies). This procedure potentially generates a large number of optimal binary resolutions. Moreover, there may be multiple minimum cost event histories for each binary resolution. The lists saved in $\mathcal{H}$ allow the algorithm to generate all optimal binary resolutions and every event history associated with the binary resolutions. Generating all solutions enables the user to choose the best resolution based on problem-specific criteria.

**Algorithm 12** Resolve-DTL algorithm

```
1 DTLResolve(V_G, V_S):
2    for each g ∈ V_G post order:
3      if g is binary:
4        for each s_1, s_2 ∈ V_S × V_S:  generateCostTable(g, s_1, s_2)
5      else:
6        for each T_g^b = (V_g^b, E_g^b) ∈ 𝔗(g):
7          DTLResolve(V_g^b, V_S)
8          Merge(H_g^b, H_g, K_g^b, K_g, T_g^b)
9 Merge(H_g^b, H_g, K_g^b, K_g, T_g^b):
10   for each s ∈ V_S :
11     if K_g^b[s] < K_g[s] :
12       clear H_g[s]
13       for each (ϵ, s_l, s_r) in H_g^b[s]:enqueue (ϵ, s_l, s_r, T_g^b) in H_g[s]
14       K_g[s] = K_g^b[s]
15     else if K_g^b[s] == K_g[s]:
16       for each (ϵ, s_l, s_r) in H_g^b[s]:enqueue (ϵ, s_l, s_r, T_g^b) in H_g[s]
```

The worst case complexity for calculating the cost table for a single binary resolution of a polytomy $g$ is $O(|V_S|^2 k(g))$. The algorithm to calculate the cost tables for all binary resolutions of this polytomy has worst case complexity of $O(|V_S|^2 k(g) N_g)$ where $N_g = (2k(g)-3)!!$ is the total number of possible binary trees with $k(g)$ leaves. The cost tables at any node, whether binary or polytomy, depend only on the cost tables at the node's children. The post-order traversal ensures that tables are calculated before they are used. Thus, calculating cost tables for a given polytomy and its binary resolutions is independent from other polytomies in the gene tree. As a result, for a tree with multiple polytomies, the total number of possible

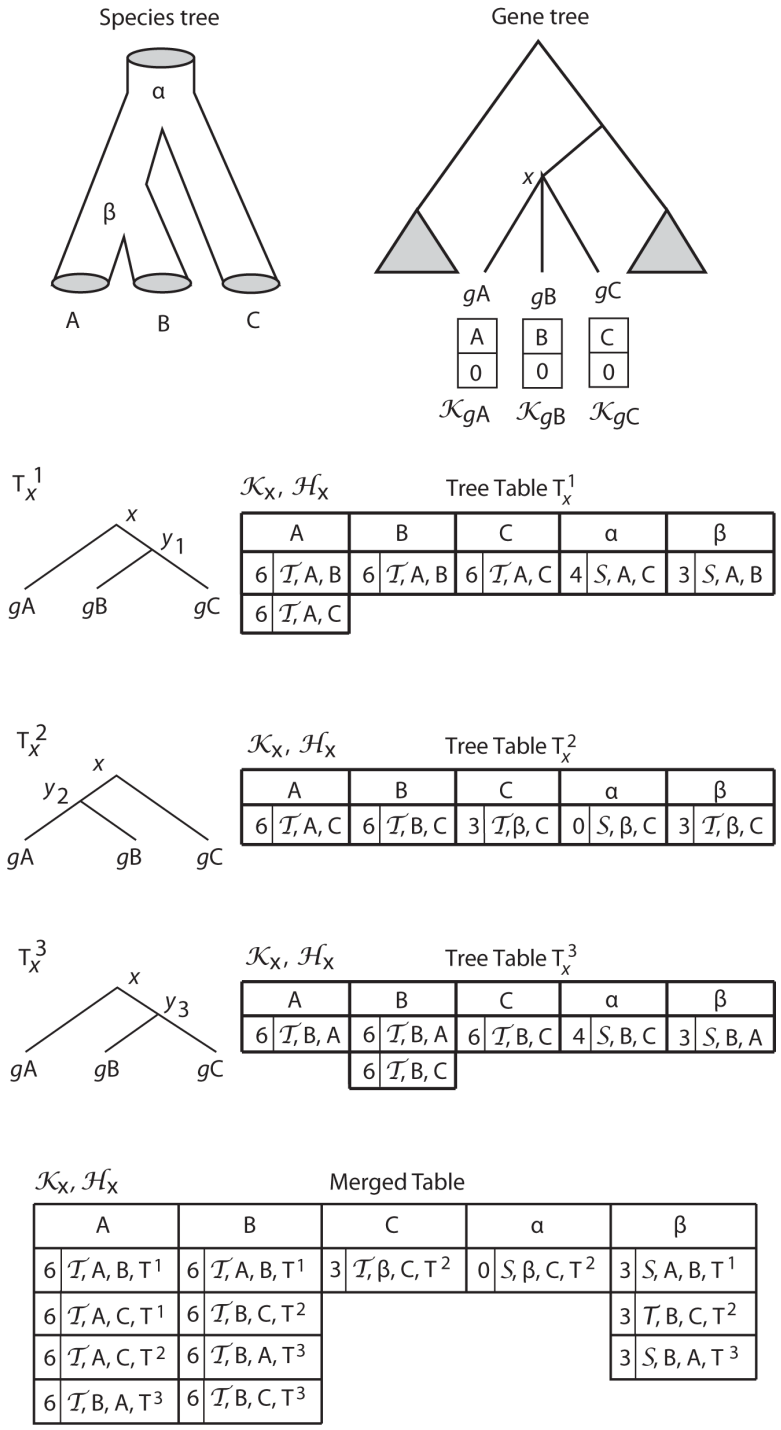



Figure 4.2: Dynamic programming to calculate cost table for a non-binary gene node $x$. This process tries different binary resolutions for node $x$; for each tree, it uses the same dynamic programming to generate cost table for $x$. Then all cost tables for all binary resolutions are merged, and the entry in tuple table is added an extra term to store which tree was used.

binary resolutions for the gene tree is $\sum_{g \in V_G \ni k_g > 2} N_g$. Because the number of possible binary resolutions increases super-exponentially with the size of the polytomy, this summation is dominated by the largest polytomy in the unresolved gene tree, and the complexity of the exact algorithm is $O(|V_S|^2 k_m N_m)$, where $N_m$ is the number of binary topologies of size $k_m = \max_g k_g$.

### 4.3.1. Heuristics for Resolve-DTL

Given an unresolved gene tree, the Resolve-DTL algorithm described in the previous section finds all most parsimonious binary resolutions with respect to a given set of event costs. However, finding the optimal solution requires enumeration of all possible binary resolutions of each polytomy in $T_G$, which is intractable for larger polytomies. For example, evaluating the 10,395 possible binary topologies for a polytomy with 7 leaves requires $\sim$ 20 seconds on a 2.4 GHz processor. For a polytomy of size 8, there are over 135k possible resolutions and the running time on the same machine is approximately 5 min. For phylogenomic analyses with thousands of gene trees, where many gene trees may contain large polytomies, heuristics become essential.

Instead of enumerating all binary resolutions, as we do in the exact algorithm, we replace $\mathfrak{T}(g)$ in line 11 of Algorithm 12 with $\widehat{\mathfrak{T}}(g)$, a subset of all possible binary resolutions. We propose several fast heuristics based on different strategies for defining this subset:

**4.3.1.0.1. NNI** A direct alternative to enumerating all binary resolutions of a polytomy is to use tree modification techniques, such as Nearest Neighbor Interchange (NNI), to sample $M$ binary resolutions from the entire search space, where the sample size $M$ is selected by the user. Given an unresolved tree wherein the polytomies are the result of collapsing weak edges in the input tree, the search starts from the original binary topology. If the input tree itself was non-binary, then a starting topology is selected arbitrarily. At each iteration, NNI is applied to a randomly selected internal branch in the current tree. If the reconciliation cost of the resulting tree is equal to the best cost seen so far, the tree is added to the list of candidate resolutions. If it is lower, the current tree replaces the list of candidate resolutions. The worst case complexity of this method is $O(|V_S|^2 k(g) M)$, for a single polytomy $g$.

**4.3.1.0.2. Local DL** In this heuristic, we resolve each polytomy by replacing it with a single binary resolution that minimizes the DL cost, rather than the DTL cost. The quality of this heuristic will depend on how well the binary resolution that minimizes the DL cost approximates the binary resolution that minimizes the DTL cost.

Implementing the DL-resolution algorithm (Durand et al., 2006b) in this context requires some modification. Recall that with DL-reconciliation, $\mathcal{M}[g]$ is calculated directly from $\mathcal{M}[g_i] \forall g_i \in C(g)$, the mappings at the children. However, under the DTL-event model, the mappings at the children of $g$ are not finalized until pass two; instead, each child is associated with tables for *possible* species assignments. Rather than attempt all combinations of mappings at the children, we assign $\mathcal{M}[g_i] = \text{argmin}_s \mathcal{K}_{g_i}[s]$, the species that minimizes the cost at $g_i$. The DL-resolution algorithm is then applied to find a binary resolution, $T_g^{DL}$ of $g$. If there is more than one minimum cost resolution, a single resolution is selected arbitrarily.

The polytomy at $g$ is then replaced with $T_g^{DL}$, a binary resolution that minimizes the DL score with these labels, resulting in $2k(g)-1$ new binary nodes. The post-order traversal then continues up the tree, calculating the entries in tables $\mathcal{K}$ and $\mathcal{H}$ at the internal nodes in $T_g^{DL}$ using the DTL-event model. Note that since this approach yields a single binary resolution, calculating $\mathcal{H}_g$ does not require the call to the `Merge` function (line 13 in Algorithm 12). Each entry in $\mathcal{H}_g$ is associated with that single binary resolution, although there may still be degeneracy if the binary resolution has more than one minimum cost event history. Under the DL-event model, the time required to find an optimal resolution of a polytomy $g$ is $O(|V_S| + k(g))$ (Lafond et al., 2014). The calculation of the cost tables for the nodes in the new binary resolution has complexity $O(|V_S|^2 k(g))$. Since this step is more computationally demanding then obtaining the DL-resolution, the worst case complexity for a single polytomy is $O(|V_S|^2 k(g))$.

**4.3.1.0.3. Hybrid DL-NNI** The quality of the results of the *NNI* strategy, described above, are likely sensitive to the tree chosen to start the sampling process. Instead of starting with the original binary branching order, or an arbitrarily chosen one, we posited that a tree that minimizes the DL-cost might be closer, in tree space, to the optimal DTL-resolution. In the *Hybrid* heuristic, we apply the *Local DL* heuristic to generate a candidate starting tree and then sample candidate resolutions in tree space by applying *NNI* for $M$ iterations. Since the complexity of *Local DL* is low compared to DTL-reconciliation, adding this extra

step does not have an adverse effect on the running time. The worst case complexity is $O(|V_S|^2 k(g)M)$ for a single polytomy.

**4.3.1.0.4. Global DL** As a comparison to the Resolve-DTL heuristics, we also calculated the optimal DL-resolution (Durand et al., 2006b; M. et al., 2012) of the entire input gene tree. Once DL-resolution is complete, the DTL-reconciliation cost of the resulting binary tree is calculated by reconciling it with the DTL-event model. This is in contrast to the *Local DL* strategy, described above, which uses the DL model to generate a binary resolution of each polytomy independently, but uses cost tables, obtained with the DTL model, as input on the children of each polytomy and constructs cost tables on each node of the binary resolution based on the DTL model.

We implemented Resolve-DTL (the *Exact* method, Algorithm 12) and the heuristic strategies described above in the NOTUNG reconciliation software package. For the *Exact* method, NOTUNG finds all optimal DTL-resolutions and calculates all DTL-reconciliations for each one. All temporally feasible solutions are included in the output, up to a user-specified limit. Temporally infeasible reconciliations are discarded. No binary resolutions are generated if all solutions are temporally infeasible.

For all heuristics (except *Global DL*), for polytomies of size $k \leq 6$, which have at most 945 binary resolutions, the *Exact* method was applied and all binary resolutions of the polytomy are enumerated. The chosen heuristic method is applied for polytomies of size $k \geq 7$.

To speed up the running time in practice, several memoization techniques are implemented. When enumerating all possible binary resolutions for a polytomy, the order of the topologies is sorted such that the left subtree is kept stable if there are more topologies to check in the right subtree. This effectively maximizes memoization because we can reuse the left cost tables when searching right topologies. Although this does not improve the complexity, this significantly reduces runtime. In addition, because each NNI operation inserts a single new branch in $T_G$, it is therefore sufficient to recalculate only the cost tables, $\mathcal{K}$ and $\mathcal{H}$, associated with nodes on that branch and its ancestors, instead of recalculating tables for the entire binary resolution.

## 4.4. Performance analysis

To investigate the behavior of these methods, these strategies were evaluated empirically on two previously published data sets:

***ALE* trees:** In order to obtain a benchmark for which events are known, we used a data set (Szöllõsi et al., 2013a,b), wherein a high confidence set of events were inferred from multiple alignments of simulated sequences using a probabilistic model that integrates sequence evolution and gene events. This dataset is based on 1099 HOGENOM gene families (Penel et al., 2009) sampled from 36 cyanobacterial genomes (Szöllosi et al., 2012). For each gene family, we obtained the gene events and the multiple alignments (MSAs) of amino acid sequences from which they were inferred (Szöllõsi et al., 2013b). We constructed "species tree blind" gene trees from these MSAs using PhyML (Guindon et al., 2010) with the LG+G+I model with four Γ-distributed rate categories. Branch support scores were obtained by bootstrapping with 100 replicates.

***Barker* trees:** Gene trees for 13,854 OrthoMCL families, sampled from 49 cyanobacteria and 16 proteobacteria, were downloaded from (Barker, 2013). The gene trees, and the associated species tree, were constructed as described in (Darby et al., 2017; Latysheva et al., 2012). Briefly, maximum likelihood trees were constructed from amino acid sequence alignments using PhyML, following model selection with ModelGenerator (Keane et al., 2006). Branch support was assessed using 200 bootstrap replicates.

Trees were processed using NOTUNG. All NOTUNG analyses were carried out with default event costs ($C_T = 3$, $C_D = 1.5$, $C_L = 1$). These transfer and loss costs are identical to those used with the same data sets in previous studies (Darby et al., 2017; David and Alm, 2011; Jacox et al., 2017). We chose a slightly lower duplication cost to reduce the number of degenerate solutions. These costs are also consistent with costs used in recent phylogenomic analyses (e.g., (David and Alm, 2011; Stolzer et al., 2012a)), which were selected to minimize the total change in genome size, averaged over the species tree (David and Alm, 2011).

Trees from both datasets were rooted using DTL-event parsimony with NOTUNG. Unresolved trees were then created by collapsing branches with bootstrap values below 70%. To assess the impact of threshold choice, the Barker dataset analyses were repeated with thresholds of 80% and 90% (Table 4.1). Gene trees lacking weak edges for a given threshold were eliminated from the analysis at that threshold.

| Threshold | $k_m > 2$ | $2 < k_m < 7$ | $k_m = 7$ | $k_m > 7$ | $N_P$ |
|---|---|---|---|---|---|
| Barker trees | | | | | |
| 70% | 3672 | 2678 | 288 | 706 | 7370 |
| 70% F | 3604 | 2650 | 281 | 673 | – |
| 80% | 4280 | 2983 | 337 | 960 | 9196 |
| 80% F | 4204 | 2955 | 329 | 920 | – |
| 90% | 4882 | 3272 | 406 | 1204 | 10307 |
| 90% F | 4770 | 3241 | 394 | 1135 | – |
| ALE trees | | | | | |
| 70% | 1080 | 383 | 116 | 581 | 5210 |
| 70% F | 813 | 346 | 90 | 444 | – |

Table 4.1: The number of unresolved gene trees as a function of branch support threshold and maximum polytomy size ($k_m$). $F$ indicates the number of trees remaining, after trees that lacked a temporally feasible solution for one or more methods were removed from consideration. $N_P$: number of polytomies of any size in all trees, combined.

The resulting rooted, non-binary trees were resolved using the *Exact*, *NNI*, *Local DL*, and *Hybrid* strategies. The trees were also resolved with the *Global DL* model for comparison. This process resulted in six trees for each gene family: five rearranged trees and the original tree (i.e., the rooted binary tree prior to collapsing weak bootstrap values). All six trees were reconciled using the DTL model in NOTUNG to obtain DTL event costs (Eqn. 2.1), which were used to assess the performance of the different binary resolution strategies. Inferred event histories were tested for temporal feasibility; only gene families for which all six trees had at least one temporally feasible reconciliation were retained for further analysis (Table 4.1).

### 4.4.1. Accuracy

The accuracy of the Resolve-DTL heuristics was evaluated using three different measures. First, for each gene family in the ALE, the event costs obtained with all methods were compared with the weighted sum of the high-confidence events associated with that family, calculated using Eqn. 4.1. Second, for families in both datasets, the event costs of binary resolutions obtained with each heuristic were evaluated relative to the event costs obtained with the *Exact* method. Third, the binary tree obtained with each heuristic was compared with the binary tree obtained with the *Exact* method, using the normalized *Robinson-Foulds (RF) distance*, a tree comparison method that is frequently used to assess tree accuracy. To

determine how well the heuristic strategies approximate the exact approach, we focused on gene trees with maximum polytomy size of $k_m = 7$. The space of trees with seven leaves is small enough to admit exhaustive search, so that the reduction in event cost obtained with the various heuristics can be compared with the optimal improvement in cost obtained with the *Exact* method.

For the cost-based comparisons, the per-family accuracy was quantified using the error,

$$\Delta(\widehat{\kappa}_H, \kappa^*) = \frac{\widehat{\kappa}_H - \kappa^*}{\kappa^*}, \tag{4.2}$$

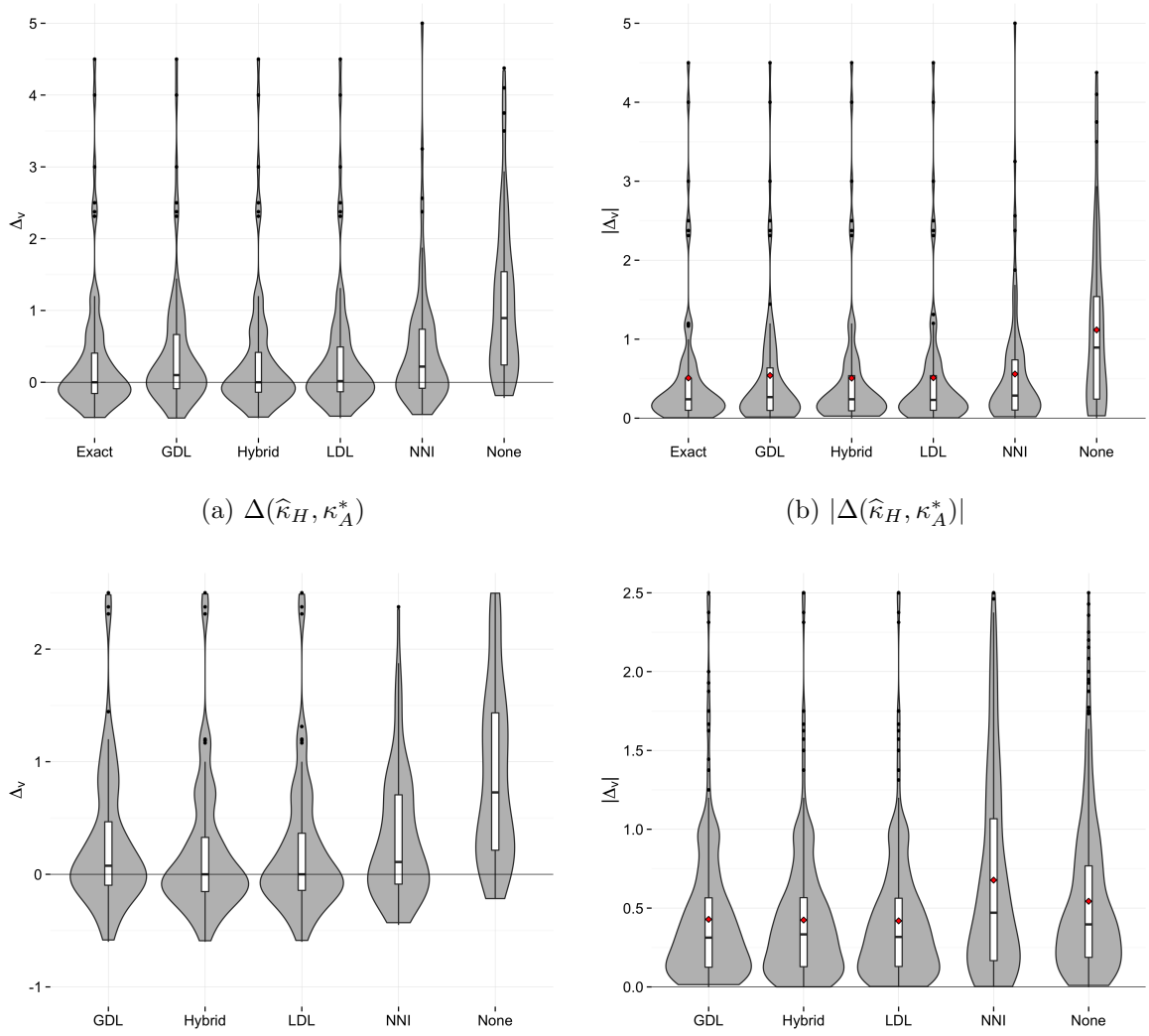
where $\widehat{\kappa}_H$ represents the lowest DTL-reconciliation cost obtained with heuristic $H$ and $\kappa^*$ is the overall best score for the family. For the ALE trees, the best score is defined to be $\kappa^*_A$, the weighted sum (Eqn. 2.1) of the high-confidence events inferred for that family. For comparison with the *Exact* method, for both sets of trees, the best score is $\kappa^*_E$, the optimal DTL-cost. Because *Exact* is guaranteed to find the minimum cost resolution under the Maximum Parsimony criterion, $\Delta(\widehat{\kappa}_H, \kappa^*_E)$ is always positive.

#### 4.4.1.1. Comparison to High-Confidence Events Using ALE Data

First, to set these methods in context, we evaluated both the exact and the heuristic methods relative to the high-confidence inferred events for each ALE family (Fig. 4.3). All strategies reduced the magnitude of the error by a factor of two, on average, relative to the uncorrected tree (Fig. 4.3b). While a small number of trees had very large errors, for most trees $|\Delta(\widehat{\kappa}_H, \kappa^*_A)| < 1.0$. The greatest reductions in error were obtained with the *Exact*, *Hybrid*, and *Local DL* strategies. All strategies underestimate the number of events in some cases (Fig. 4.3a). The strategies for which the magnitude of the error is lowest, *Exact*, *Hybrid*, and *Local DL*, underestimate the number of events roughly half the time. *Global DL* and *NNI* have a greater tendency to overestimate than to underestimate the events.

#### 4.4.1.2. Comparison of heuristic methods with the Exact method:

Next, we assessed how well various heuristics approximate the *Exact* algorithm for both the ALE trees and the Barker trees. Both datasets display similar trends (Fig. 4.4a), and similar behavior was observed with thresholds of 80% and 90% (Fig. 4.5a).



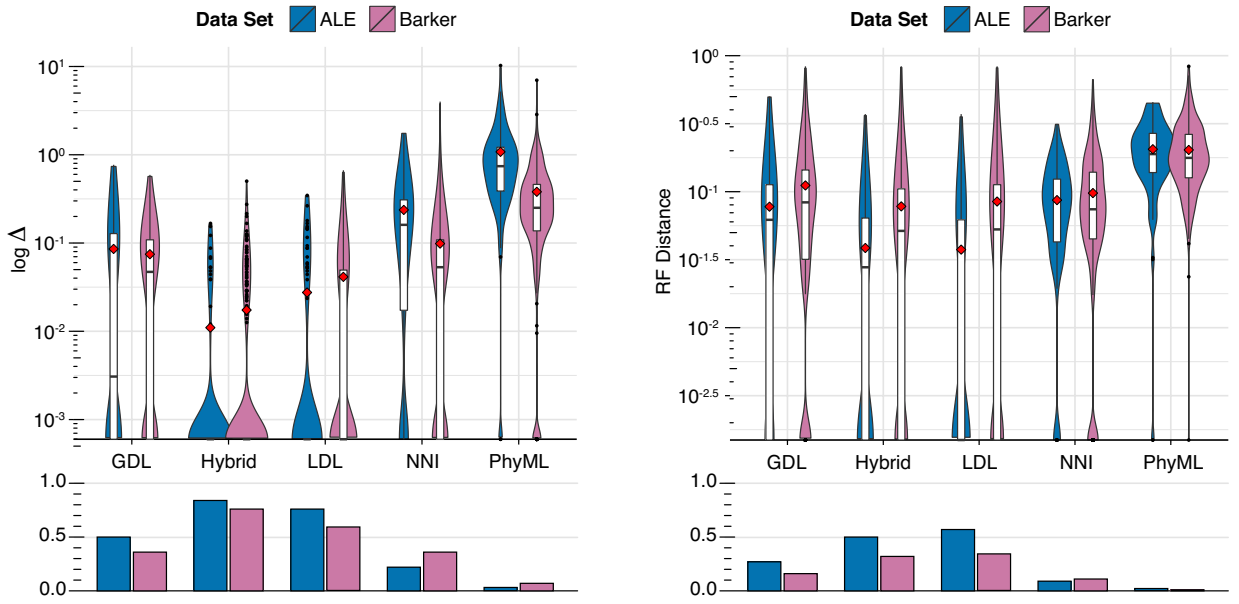

(a) $\Delta(\widehat{\kappa}_H, \kappa_A^*)$

(b) $|\Delta(\widehat{\kappa}_H, \kappa_A^*)|$

(c) $\Delta(\widehat{\kappa}_H, \kappa_A^*)$ with larger polytomies

(d) $|\Delta(\widehat{\kappa}_H, \kappa_A^*)|$ with larger polytomies

Figure 4.3: Error of the DTL reconciliation score obtained with various heuristics on the simulated data set. Error was calculated as a function of the weighted sum of the ALE events. (a) Error for gene trees with polytomy of size 7. (b) Absolute value of the error for gene trees with polytomy of size 7. (c) Error and (d) absolute value of the error for gene trees with at least one polytomy of size greater than 7. The threshold to collapse weak branches is 70%, and the number of NNI samples used was 1000.




(a) $\log \Delta(\widehat{\kappa}_H, \kappa_E^*)$ (b) $\log \Delta_{\mathrm{RF}}$

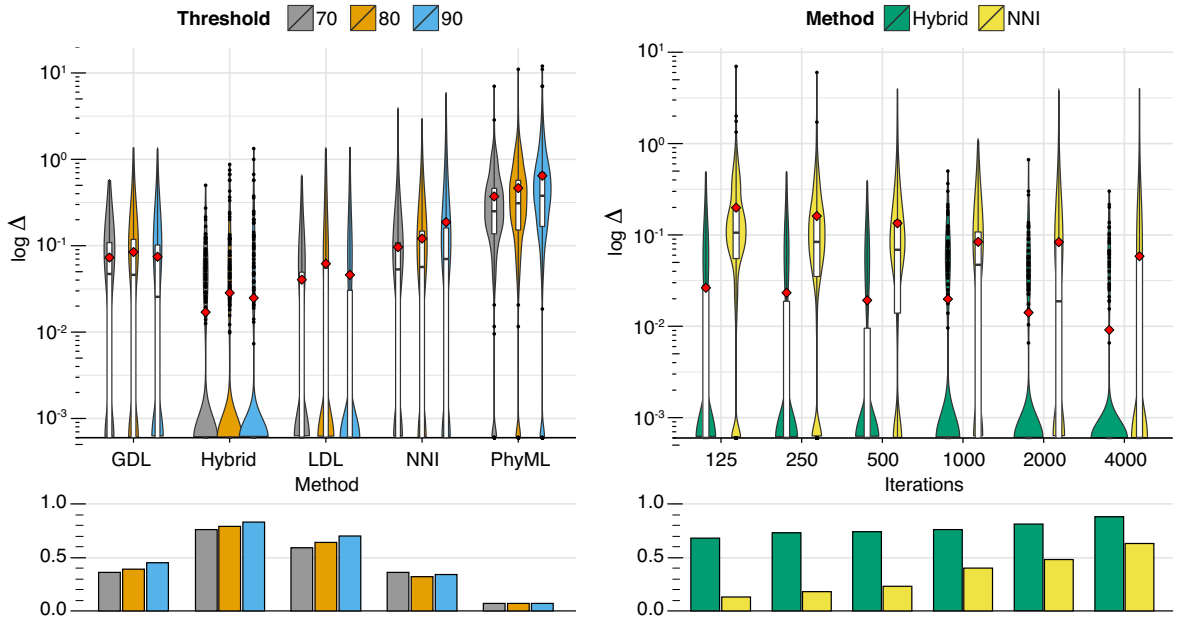
Figure 4.4: Comparison of heuristic strategies with the *Exact* method for ALE and Barker trees, with a branch support threshold of 70%. The *NNI* and *Hybrid* strategies were executed with $M = 1000$ samples. Red points indicate the mean. *(a)* Top: $\log \Delta(\widehat{\kappa}_H, \kappa_E^*)$. Zero values are represented as 0.01 times the minimum non-zero error. Bottom: the proportion of trees where $\Delta(\widehat{\kappa}_H, \kappa_E^*) = 0$. *(b)* Top: Log normalized RF distance ($\log \Delta_{\mathrm{RF}}$). Bottom: the proportion of trees where $\Delta_{\mathrm{RF}} = 0$.

The mean, median and maximum costs obtained with all Resolve-DTL strategies are markedly lower than the corresponding costs obtained with the uncorrected PhyML trees. When no correction is applied, $\Delta(\widehat{\kappa}_H, \kappa_E^*) = 0$ for only 7% of Barker trees and 2% of ALE trees. Thus, in almost all cases, when the branch support was less than 70%, it was possible to find an alternate binary topology that agrees with all strongly supported branches and reduces the number of gene events required to explain the data.

Substantially better results are obtained with *Local DL* and *Hybrid* than with the other heuristic strategies. In fact, the *Hybrid* strategy performed as well as the *Exact* method for 84% of ALE trees and 74% of Barker trees. Heuristics based on the DL-event model performed surprisingly well, given that transfers are likely common in bacterial gene family evolution. For almost half of ALE trees, the *Global DL* strategy found a binary resolution with a DTL-reconciliation cost as low as the optimal, *Exact* cost. The *Local DL* heuristic

performed well for both data sets; 75% of ALE trees and 58% of Barker trees obtained optimal DTL-reconciliation costs.

#### 4.4.1.3. Impact of Different Numbers of Iterations




(a) $\log \Delta(\widehat{\kappa}_H, \kappa_E^*)$ (b) $\log \Delta(\widehat{\kappa}_H, \kappa_E^*)$

Figure 4.5: Comparison of heuristic strategies with the *Exact* method for the Barker trees only. Red points indicate the mean. *(a)* Performance with branch support thresholds of 70%, 80%, and 90%. The *NNI* and *Hybrid* strategies were executed with $M = 1000$ samples. Top: $\log \Delta(\widehat{\kappa}_H, \kappa_E^*)$. Zero values are represented as 0.01 times the minimum non-zero error. Bottom: the proportion of trees where $\Delta(\widehat{\kappa}_H, \kappa_E^*) = 0$. *(b)* Performance of the *NNI* and *Hybrid* strategies with sample sizes from $M = 125$ to $M = 4000$; branch support threshold of 70%. Top: $\log \Delta(\widehat{\kappa}_H, \kappa_E^*)$. Zero values are represented as 0.01 times the minimum non-zero error. Bottom: the proportion of trees where $\Delta(\widehat{\kappa}_H, \kappa_E^*) = 0$.

The *NNI* and *Hybrid* strategies use tree modification techniques to search for a low cost resolution. Two factors that may influence the effectiveness of this strategy are the tree used as the starting point in the search and the number candidate trees that are sampled. The *Hybrid* strategy substantially outperforms *NNI* with $M = 1000$ samples (Figs. 4.3 and 4.4a), demonstrating that the starting state is, indeed, an important factor in the quality of the outcome.

| Iterations | % of Trees ($k_m = 7$) |
|---|---|
| 125 | 1.2 |
| 250 | 2.4 |
| 500 | 4.8 |
| 1000 | 9.6 |
| 2000 | 19.2 |
| 4000 | 38.4 |

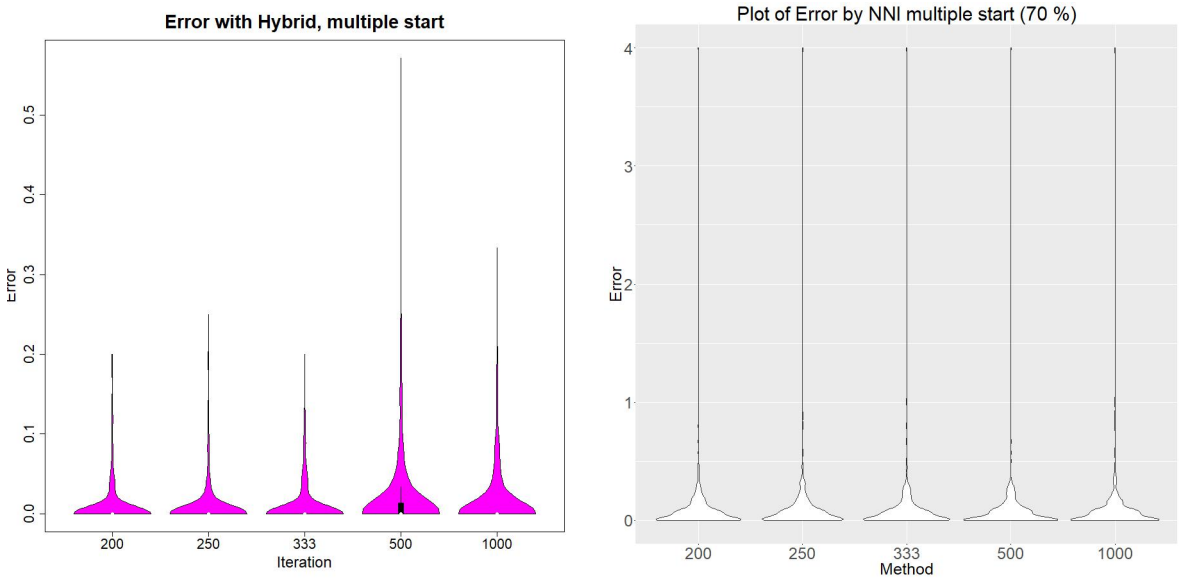
Table 4.2: Experiments with various number of iterations for NNI and Hybrid. The second column indicates the percentage of total resolutions of a polytomy of size 7 that is covered by the number of iterations.

| Number Starts | Iterations per Start |
|---|---|
| 1 | 1000 |
| 2 | 500 |
| 3 | 333 |
| 4 | 250 |
| 5 | 200 |

Table 4.3: Number of starts for each experiment. The total number of iterations is set to 1000. The first experiment starts NNI once for 1000 iterations, the second experiment start NNI twice, each for 500 iterations, etc.

To assess how the accuracy depends on the number of topologies sampled, we repeated the analysis of the Barker trees with the *NNI* and *Hybrid* strategies with sample sizes ranging from $M = 125$ to $M = 4,000$. Predictably, the error decreases as the number of samples increases for both strategies (Fig. 4.5b). The *Hybrid* strategy performs as well as the *Exact* strategy for three out of four trees, when $M = 1000$. The marginal gain in accuracy obtained with sample sizes greater than 1000 is limited. In contrast, for *NNI*, the mean and median error continue to improve over the entire range. Moreover, even when $M = 4000$, more than 37% of trees still have non-zero error. A sample size of 4000 corresponds to almost 40% of binary topologies with $k = 7$ leaves. With this sample size, *NNI* does not represent a dramatic reduction in computational load relative to exhaustive search, yet almost half the trees could be further improved. With larger polytomy sizes, the fraction of tree space that can be sampled in a realistic time frame will dwindle. For example, for a polytomy of size eight, 4000 NNI modifications will sample less than 3% of possible topologies.

#### 4.4.1.4. Multiple Runs of NNI and Hybrid




(a) $\Delta(\widehat{\kappa}_H, \kappa_E^*)$, Multiple starts, Hybrid

(b) $\Delta(\widehat{\kappa}_H, \kappa_E^*)$, Multiple starts, NNI

Figure 4.6: $\Delta(\widehat{\kappa}_H, \kappa_E^*)$ for multiple runs of the iteration-based methods, (a) Hybrid and (b) NNI, using Barker data set. Each method was run multiple times, with a total of 1000 iterations.

The hybrid and NNI methods randomly search tree space to find the best binary resolution. At each step, an NNI tree is chosen at random, Thus, the random walk may be unpredictable. Poor moves at the beginning of the search may lead to searching in a region of tree space that does not have lowest cost. To better understand the implications of random moves on finding the lowest-cost rearrangement, I also set up an experiment to understand how multiple starting directions would change the result of the heuristics. Specifically, I considered how error changes when performing the NNI or hybrid method with 1,000 iterations total, but restarting the search a variable number of times (Table 4.3). The results are shown in Figure 4.6. For the *Hybrid* strategy at 1000 iterations, the error was already lower compared to other heuristics. With multiple runs of fewer iterations, the mean error can be further reduced. This suggests that, for this data set, searching around the resolution found by *Local DL* is a better strategy than randomly walking a longer distance.

In contrast, for *NNI*, multiple starts did not seem to have any effect on the error. It is important to note that by restarting with the resolution first proposed, this approach will

cover more of the tree space surrounding the first resolution, and less of the space distant from that resolution. For NNI, the first resolution is the original binary branching pattern, which may not be close to the best resolution. Multiple starts from the same topology may not result in a lower-cost binary resolution (Figure 4.6b). For the hybrid approach, the first resolution is the one determined by *Local DL*; the assumption is that the best binary resolution is close to the binary resolution generated by the DL strategy. Therefore, multiple starts may search a "good" area more thoroughly than a single run with many iterations (Figure 4.6a).

As the *NNI* approach starts from the original binary resolution, the best resolution may be too far from the original resolution. Therefore, the *NNI* method is likely to find better resolution by walking a longer distance.

#### 4.4.1.5. Robinson-Foulds Distance versus Cost Error

We also compared the resolved binary trees obtained with the various strategies using normalized RF distance ($\Delta_{\mathrm{RF}}$), a measure of the difference in the topologies of two binary trees with the same leaf labels. $\Delta_{\mathrm{RF}}$ is the number of bi-partitions that are present in one tree, but not the other, normalized by the total number of bi-partitions. For each family in each data set, we calculated $\Delta_{\mathrm{RF}}(T_H^b, T_E^b)$, the normalized RF distance between the binary resolution obtained with heuristic $H$ and the binary resolution obtained with the *Exact* method (Fig. 4.4b).

The overall trends observed with $\Delta_{\mathrm{RF}}$ are similar to those observed with $\Delta(\widehat{\kappa}_H, \kappa_E^*)$. Most uncorrected PhyML trees display some dissimilarity with the *Exact* binary resolution. For corrected trees, the binary resolutions obtained with the *Local DL* and *Hybrid* strategies are most similar to the *Exact* binary resolution.

However, the shapes of the $\Delta_{\mathrm{RF}}(T_H^b, T_E^b)$ and $\Delta(\widehat{\kappa}_H, \kappa_E^*)$ distributions are dissimilar (Fig. 4.4), suggesting that for individual gene families, the two measures do not reflect the same information. This is further supported by the Pearson correlation coefficients of $\Delta_{\mathrm{RF}}(T_H^b, T_E^b)$ and $\Delta(\widehat{\kappa}_H, \kappa_E^*)$ for the various strategies (Table 4.4). The strongest correlation observed for any method in either dataset is 0.32. The other correlations are even weaker and only half are significant at the 0.005 level.

Several factors may be contributing to the lack of correlation between $\Delta(\widehat{\kappa}_H, \kappa_E^*)$ and $\Delta_{\mathrm{RF}}$.

First, degeneracy may inflate the reported RF distance. If there are multiple optimal binary resolutions, but only one, chosen arbitrarily, is reported by each method, then a non-zero RF distance can result, even if both methods find the same set of binary resolutions. Reconciliation cost is not sensitive to degeneracy in this way. If a heuristic method finds at least one optimal resolution, then $\Delta(\widehat{\kappa}_H, \kappa_E^*)$ will be zero, since it depends on the event cost, not the tree topology.

Recall that the heuristic strategies are not guaranteed to return all optimal binary resolutions. The *Global DL* and *Local DL* strategies will find one binary resolution, which may, or may not, be optimal. The number of optimal binary resolutions found by the *NNI* and *Hybrid* strategies will depend on the sampling process. In fact, Table 1 shows that for roughly 20% to 40% of trees, $\Delta(\widehat{\kappa}_H, \kappa_E^*) = 0$, but $\Delta_{\mathrm{RF}}(T_H^b, T_E^b) \neq 0$. This suggests that the fact that the heuristics only find a subset of the binary resolutions is contributing to the difference between error and RF distance.

Even when all resolutions found by the heuristics are compared against all optimal resolutions found by the *Exact* method, event-cost error and RF distance may be poorly correlated because they capture different information. A pair of topologies with similar event histories may have few bi-partitions in common and vice versa. For example, the trees $(A,(B,(C,(D,E))))$ and $(B,(C,(D,(A,E))))$ differ by a single transfer, but have no bi-paritions in common. Zheng and Zhang (2014b) recently showed that RF distance is not equivalent to event-based distances in a DL model. This is likely also true for event distances based on transfers.

| Method | Barker | | ALE | | Barker, $\Delta = 0$ | | ALE, $\Delta = 0$ | |
|---|---|---|---|---|---|---|---|---|
| | r | P-value | r | P-value | All | $\Delta_{\mathrm{RF}} \neq 0$ | All | $\Delta_{\mathrm{RF}} \neq 0$ |
| LDL | 0.13 | 0.214 | 0.18 | 0.001 | 0.58 | 0.24 | 0.75 | 0.20 |
| GDL | 0.17 | 0.091 | 0.21 | 0.000 | 0.35 | 0.20 | 0.48 | 0.20 |
| NNI | 0.23 | 0.022 | 0.08 | 0.145 | 0.32 | 0.25 | 0.23 | 0.16 |
| Hybrid | 0.14 | 0.159 | -0.01 | 0.906 | 0.74 | 0.42 | 0.84 | 0.35 |
| PhyML | 0.32 | 0.001 | 0.19 | 0.001 | 0.07 | 0.06 | 0.03 | 0.02 |

Table 4.4: Correlation between RF and Error for ALE and Barker data compare to Exact Tree

### 4.4.2. Runtime

I recorded time to find a binary resolution for each polytomy of size $k < 8$ for the *Exact* method, and $k > 6$ for the heuristics (Figure 4.7). The runtime for the exact method indeed increases dramatically with the polytomy size. The *NNI* and *Hybrid* method show very similar trends to each other, and, as expected, their runtime is 1000 times greater than that of *Local DL*. For $k = 6$, the *Exact* method considers 945 binary resolutions, which is comparable to the 1000 iterations used in *NNI* and *Hybrid*. This is reflected in the similar runtimes for the three methods, suggesting that all three methods take the same amount of time to recalculate the cost tables for each binary resolution. Thus, the major factor for runtime is the number of binary resolutions sampled. Recall that the worst case complexity

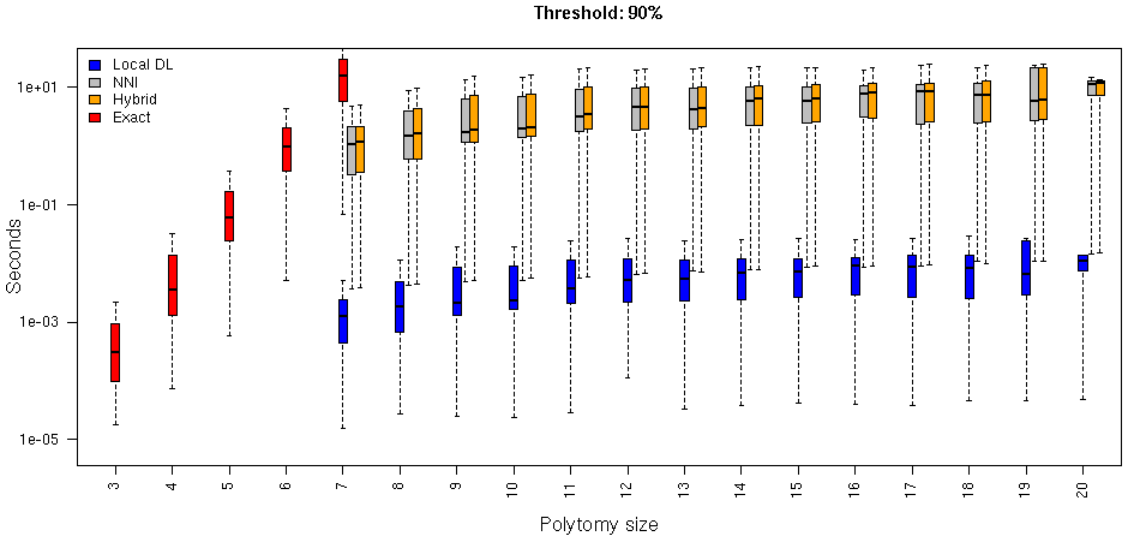



Figure 4.7: Runtime, per polytomy of the *Exact*, *Local DL*, *NNI*, and *Hybrid* strategies. *NNI* and *Hybrid* with $M = 1000$ samples. The x-axis indicates the polytomy of different sizes, and the y-axis represents the log of runtime in seconds.

for resolving a polytomy of size $k$ with *Hybrid* and *NNI* is $O(|V_S|^2 kM)$. Since $M$ remains constant, I might expect runtime to vary with polytomy size $k$. However, in Figure 4.7, the runtime of *Hybrid* and *NNI* are almost flat as polytomy size increases. This could be caused by memoization that significantly reduced the number of new nodes for which cost tables must be recalculated, or that the runtime for these experiments are not in the asymptotic regime of the big-O complexity.

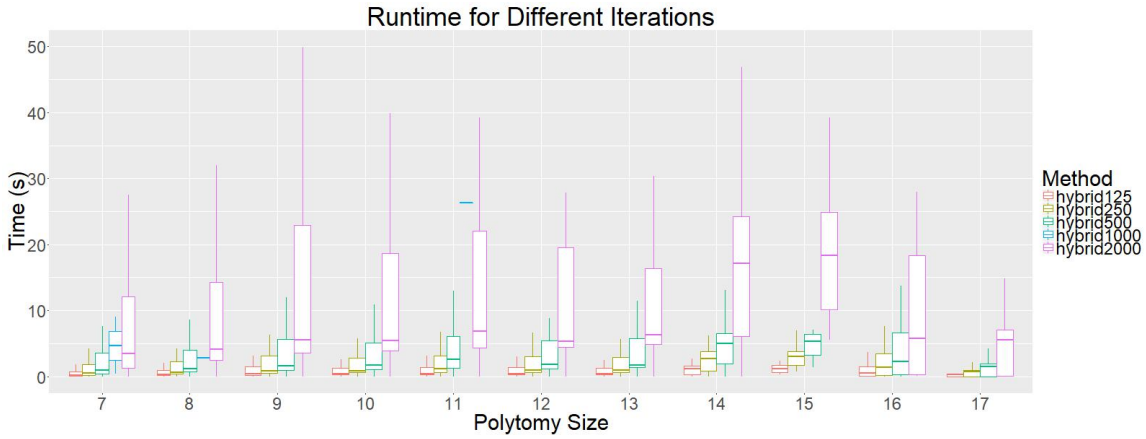
I also observe large variance on the runtime for the same polytomy size for each method.

This is because calculating the cost table for a gene node may not always be the worst case complexity ($O(|V_S|^2)$). Recall in DTL reconciliation, entries in $\mathcal{H}$ and $\mathcal{K}$ are calculated by considering all possible combinations for the mappings at $l(g)$ and $r(g)$. The runtime of this calculation could vary drastically based on the number of possible labels on the left and right children. For example, if $l(g)$ and $r(g)$ are leaf gene nodes, which can only be mapped to the one species node—the species where this gene was sampled—then calculating the cost table for $g$ would take minimal time. In that case, the number of possible species labels for $g$ is slightly more than the number for its children. This suggests a general trend that the number of possible species labels for a gene node increases as the gene node is further from leaf.

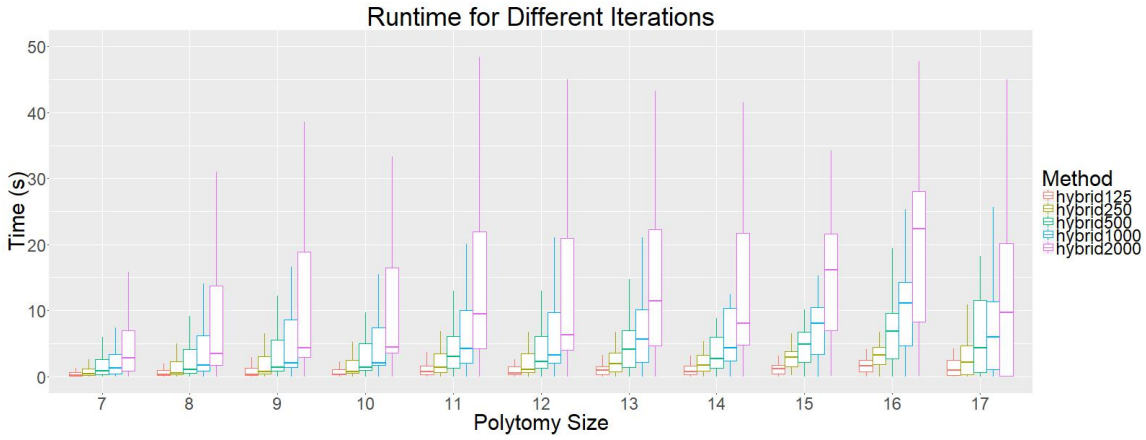
Because complexity for some heuristics depends on the number of samples, I also recorded the time for each polytomy when running *Hybrid* and *NNI* with different numbers of iterations (Figure 4.8 and 4.9). I observe that the average runtime for both *Hybrid* and *NNI* are linearly dependent on the number of iterations. The large variance of a method using the same number of iterations and for polytomies of the same size are also observed. Again this reflects the fact that calculating the cost tables for a gene node depends on the size of the cost tables on the two child nodes. Even with polytomies of the same size and sampling the same number, the same algorithm may still take various time to complete depending on the size of the cost tables on the children of the polytomy. All the algorithms that were described above were implemented in NOTUNG. The GUI is also extended to handle aspects specific to these reconciliation models. I focus on the two step corrective approach to resolve the gene tree with pre-built gene trees with uncertain branches, instead of embedding the algorithm in gene tree building software. This strategy is preferred because it can be applied to gene trees that were previously constructed for other purposes. It also provides intermediate results allowing human intervention and exploratory analyses. This approach not only generates a resolved binary gene tree topology, but also outputs the inferred events by finding the "best" binary resolution of polytomies.

## 4.5. Discussion

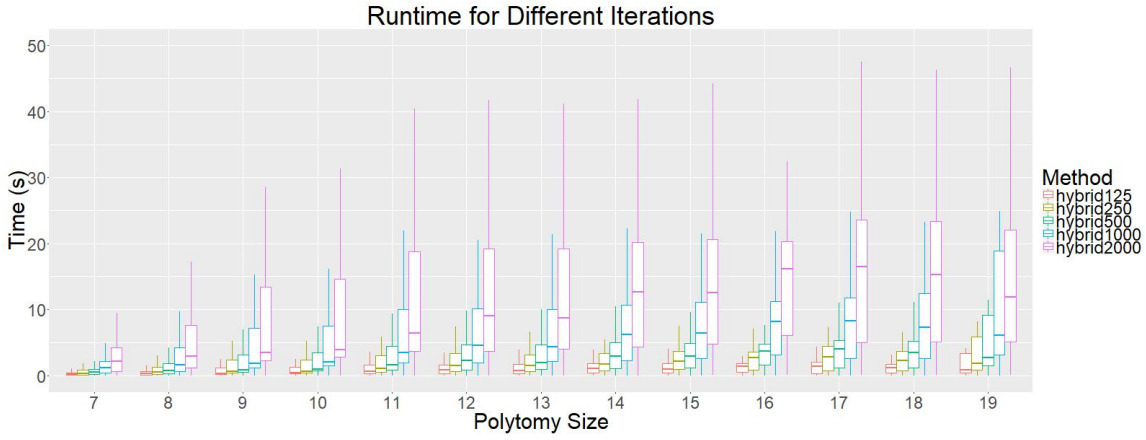
Resolving non-binary gene trees with a binary species tree under the DTL model requires an enumeration of all binary resolutions of each polytomy, resulting in a super-exponentially increasing search space. Here, we presented an exact algorithm and several promising heuris-




(a) 70% threshold

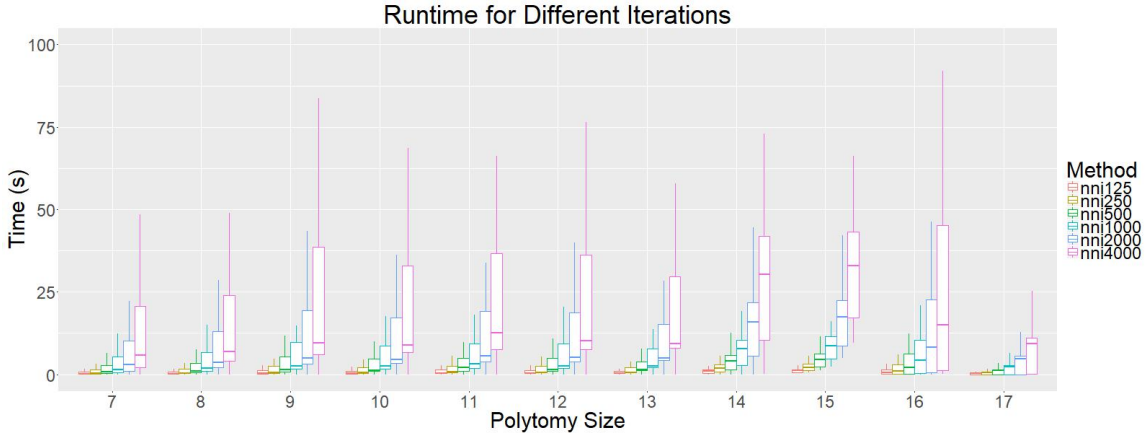



(b) 80% threshold

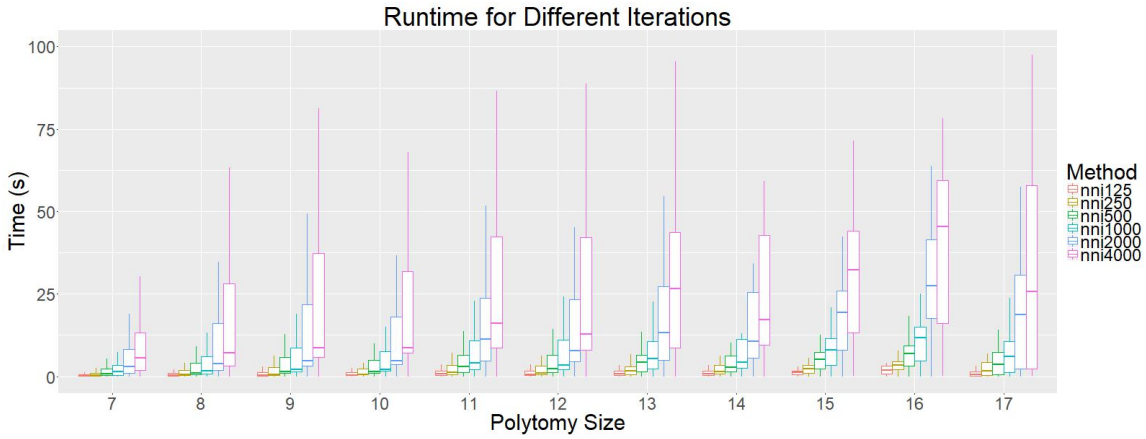



(c) 90% threshold

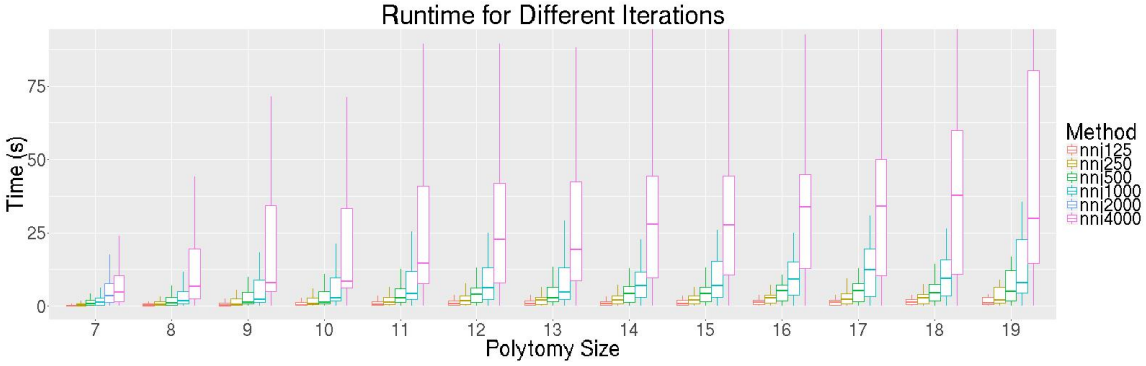
Figure 4.8: Runtime of hybrid strategy with different iteration numbers for three different thresholds.




(a) 70% threshold




(b) 80% threshold




(c) 90% threshold

Figure 4.9: Runtime of NNI strategy with different iteration numbers for three different thresholds.

tics for resolving large polytomies, which can be used as stand-alone tree correction methods or as subroutines in species tree-aware gene tree construction techniques. The algorithms are all implemented in a publicly available, Java-based software package, NOTUNG, that generates all most parsimonious resolved gene trees that are temporally feasible.

Our experimental analysis of these methods gives insights into the structure of the Resolve-DTL problem. Some heuristics are based on searching the tree space using NNI. The improvement in accuracy that can be obtained by increasing the sample size of NNI is limited. However, a well picked starting topology, combined with a modest number of iterations proved to be a good compromise in this context. Two strategies based on a DL-event model performed better than expected when applied to bacteria data sets, which likely contain horizontal transfer events, suggesting that even in the presence of transfers, the DL-model has some utility as a heuristic.

These preliminary results introduce interesting questions that could be investigated further, such as a better way to sample the search space. Using NNI, the newly proposed gene tree is nearly identical to the previous tree. As a result, NNI may be too fine grained when searching for resolutions where horizontal gene transfer can introduce more dramatic changes to the topology. Subtree Prune and Regrafting (SPR) moves propose larger jumps between topologies in tree space, which may mimic transfer moves more realistically (Hill et al., 2010). Thus, heuristics based on SPR, and possibly other tree modification procedures, are good directions for future research. Also, instead of randomly sampling the tree space, a hill-climbing approach guided by a figure-of-merit could be employed when considering proposed topology modifications.

We assessed the performance by comparing DTL-event costs obtained using each heuristic with the minimum DTL-event cost obtained using the exact method. We also calculated the Robinson-Foulds (RF) distance between the resolved trees obtained with the heuristic and exact methods and observed poor correlation between the two measures. Prior studies have shown that several metrics based on the DL model are not equivalent to RF distance (Zheng and Zhang, 2014b). Our empirical observations suggest that a theoretical comparison of measures based on DTL-events and RF distances may be a similarly fruitful direction.

Finally, this class of "species-tree" aware gene tree correction methods assumes that incongruence is due to either uncertainty or gene events. Yet other processes, such as gene conversion, incomplete lineage sorting, migration, hybridization, and introgression, can also

result in gene tree incongruence. Thus, if branches in the gene tree are weak, but reflect the true evolutionary history, tree correction methods could introduce error, rather than decrease it. Methods that account for these processes are an important direction for future work.

# Chapter 5

# Selecting Cost Using Maximum Likelihood

DTL reconciliation under maximum parsimony (MP) optimization requires a cost vector to infer gene events. The reconciliation algorithm selects the solution(s) that minimize(s) the sum of the number of occurrences of each event, weighted by its corresponding cost:

$$C(N_D, N_T, N_L) = N_D \cdot C_D + N_T \cdot C_T + N_L \cdot C_L, \tag{5.1}$$

where $C_\epsilon$ represents the weight for event type $\epsilon$, and $N_\epsilon$ is the number of events of type $\epsilon$.

The costs for duplication, transfer, and loss events are important parameters that must be set for MP reconciliation. Different costs may significantly impact the final, most parsimonious solution. Therefore, correctly selecting the cost vector ($\vec{C}$=$< C_T, C_D, C_L >$) is an important problem. In this chapter, I try to answer the question of how to select $C_\epsilon$ to obtain accurate reconciliations. I compare Maximum Likelihood (ML) reconciliation with Maximum parsimony, and discuss the possibility of selecting a cost vector based on event rates such that the most parsimonious reconciliation is equivalent to the most likely reconciliation.

Comparing ML and MP has long been studied for a related problem: evolutionary tree reconstruction. In the context of tree reconstruction, several studies have tried to answer the question: Is there a condition under which ML and MP are equivalent for building a tree from character data? Farris (1973) showed that under some reasonable assumptions, the reconstructed trees under ML and MP are equivalent. However, Felsenstein (1973) later presented a counter-example and argued that the methods in use at that time for tree

building do not satisfy the assumptions made by Farris. Nevertheless, the comparison of ML and MP in tree reconstruction still can provide valuable information of the structure of the problem.

## 5.1. Formulation of Maximum Likelihood

The gene tree reconciliation problem under a probabilistic framework has been considered in several studies. Arvestad et al. (2003) formulated the problem of probabilistic reconciliation with duplication and loss events. Later, Sjőstrand et al. (2014) introduced a DLTRS (Duplication, Loss, Transfer, Rate, and Sequence evolution) probabilistic model that incorporates gene events and sequence evolution in a single comprehensive model that infers the gene tree and the reconciliation. This method uses a modified Birth-Death event process to calculate the likelihood of observing a particular gene tree in the species tree in a model that allows genes to jump between species branches (allows transfers), while also considering sequence evolution. The formulation of likelihood in DLTRS is based on the work of Akerborg et al. (2009), Arvestad et al. (2003) and Arvestad et al. (2009), who formulated the likelihood of tree reconciliation with duplication and loss events using a Birth-Death model. These studies were focused more on tree reconstruction using a sequence evolution model, while also considering gene tree reconciliation. Khan et al. (2016) later adapted this framework to develop a model that infers gene events given a gene tree and a species tree as input. Another study formulated the probability of reconciliation with duplication and loss events under a constant event rate model instead of a Birth-Death model (Górecki and Eulenstein, 2014). Motivated by these studies, I also formulate several likelihood functions using strategies similar to Arvestad et al. (2009) under several different assumptions. In each case, I derive an estimate of the likelihood and compare the likelihood function to the cost function used in MP reconciliation.

The general ML principle is to find a parameter that maximizes the likelihood:

$$\mathcal{L} = P(X|\theta), \tag{5.2}$$

which is the probability of the observed data, $X$, given a parameter set $\theta$. The likelihood of reconciliation is slightly different. It introduces a latent variable, $\gamma$, as a reconciliation solution. Given a gene tree $T_G = (V_G, E_G)$, a species tree $T_S = (V_S, E_S)$, and a mapping

from the leaves of $T_G$ to the leaves of $T_S$, a solution can be represented as a mapping $\mathcal{M} : g \to s, \forall g \in V_G, s \in V_S$. From this mapping of gene nodes to species nodes, the gene event history can be reconstructed, as described in Chapter 2.

ML reconciliation seeks the reconciliation $\gamma$ that maximizes the probability of the observed data (i.e. the gene tree $T_G$) and the hidden latent variable, $\gamma$. In order to estimate the likelihood of a gene tree in a reconciliation framework, we require the length of each species tree branch, in units of time, and the rates of the gene events in the event model. In contrast, for MP reconciliation, events are inferred from topological differences between the gene and species tree; the branch lengths of the species tree are not required. In this chapter, we consider species trees with the branch lengths in units of time. The length of a branch $(p(s), s)$ is denoted $t_s$. The event rate vector, $\vec{r}$, contains the transfer rate ($r_T$), duplication rate ($r_D$), and loss rate ($r_L$). Given the event rates and the species tree ($T_S$) with branch lengths, the likelihood function is:

$$\mathcal{L} = P(T_G, \gamma | \vec{r}, T_S), \tag{5.3}$$

where $T_S$ is treated as a fixed parameter (Akerborg et al., 2009). In other words, Eq. 5.3 calculates the probability of a gene starting from the root of the species tree, followed by the series of gene events implied by the reconciliation solution, $\gamma$, resulting in the gene tree $T_G$. Since gene tree $T_G$ is completely specified by species tree $T_S$ and the event history implied by $\gamma$, Equation 5.3 can be restated in a simpler form:

$$\mathcal{L} = P(\gamma | \vec{r}, T_S). \tag{5.4}$$

The most likely reconciliation is then:

$$\gamma^* = argmax_\gamma P(\gamma | \vec{r}, T_S), \tag{5.5}$$

where $\gamma^*$ is the optimal reconciliation.

To estimate the probability of a reconciliation $\gamma$, we need to capture the behavior of gene events in a probabilistic framework. Specifically, we require an event process that models the probability of a gene event occurring in elapsed time $t$. In this chapter, we consider two models of the event processes through which events occur: a Constant model (C) and a Birth-Death (BD) model. As gene duplication and transfer events are both events that

can increase the gene copy number by 1. In both models, a gene duplication and a gene transfer both are considered to be a birth event, which is also a called gain event. One can calculate the probability of observing $N_D$ duplications and $N_T$ transfers by calculating the probability of $N_B$ birth events first, and then adjusting for the probability that $N_D$ of those events are duplications and the rest are transfers. This adjustment is discussed in detail in later sections.

The gene event number and gene copy number are often estimated in the probabilistic framework in later sections. One straightforward relationship between them, which is used frequently, expresses the number of gene copies after time $t$, $n(t)$, as a linear function:

$$n(t) = N_B(t) - N_L(t) + n(0), \tag{5.6}$$

where $N_B(t)$ and $N_L(t)$ denote the number of births, and losses.

### 5.1.1. Constant Event Process

The constant rate model assumes that gene event rates are independent of the number of gene copies; thus, each event rate is kept constant throughout the species tree. This assumption is appropriate in gene families where there is selection against multiple copies of the same gene. In this context, the number of events in a time interval can be modelled by a Poisson distribution and the time intervals between events are exponentially distributed.

### 5.1.2. Birth-Death Event Process

Another commonly used model for gene events is the Birth-Death (BD) process, which was originally used to model the behavior of a population (Kendall, 1949). At any time, two events may occur: A birth event increases the population size by one and a death event decreases the population size by one. The time interval before the next event occurs is sampled from an exponential distribution, given the event rate. This model can also be applied to gene event evolution, where a birth event represents a duplication or transfer that increases the gene number by one, and a death event represents a gene loss that decreases the gene number by one. One important property of the BD model is that the rate of gene events *per gene* is kept constant. As a result, the overall birth and death rates increase as

the number of gene copies increases. For example, suppose a gene family starts with one copy. After a birth event occurs, the gene copy number is increased to two and the rate of a gene event is also doubled. This is realistic in the sense that more copies of the gene should increase the chance of an event occurring on these genes. However, if the birth rate is greater than the death rate, the expected number of gene copies grows exponentially. If the birth rate is less than the death rate, the number of gene copies will drop to zero and the family will be extinguished. It is usually assumed that the death rate is not greater than birth rate. This suggests that when using this model, we implicitly assume that the growth of gene family has not reached the point where there is selection against increasing gene family size.

There are two mathematical approaches to modeling the BD process: 1) a deterministic BD process, where the population size after a time step is completely determined by the current state, and 2) a stochastic BD process, where the state of the next time step depends on the current state and events that are randomly generated based on event rates.

The time-dependent behavior of the population size in a Birth-Death model is well known (Kendall, 1949). Let $r_B$ and $r_L$ be the rate of birth and death per gene copy. In the deterministic Birth-Death model, $n(t)$, the population size after elapsed time $t$, can be directly calculated. Assuming the initial population size, $n(0)$, is 1, the population size at time $t$ satisfies the differential equation

$$\frac{dn}{dt} = (r_B - r_L)n(t). \tag{5.7}$$

Solving this differential equation yields

$$n(t) = e^{(r_B - r_L)t} \tag{5.8}$$

In the stochastic BD model, the population size after time $t$ follows a modified geometric distribution (Kendall, 1949). With an initial population size $n(0) = 1$, the probability that the population is extinct after time $t$ is

$$P(n(t) = 0|t) = \begin{cases} \dfrac{r_B e^{(r_B - r_L)t} - r_B}{r_B e^{(r_B - r_L)t} - r_L}, & \text{if } r_B \neq r_L \\ \dfrac{r_B t}{1 + r_B t}. & \text{if } r_B = r_L \end{cases} \tag{5.9}$$

The probability that the population is of size $x > 0$ is the standard geometric (Kendall,

1949):

$$P(n(t) = x|t) = (1 - \eta_t)\eta_t^{x-1}, \tag{5.10}$$

where

$$\eta_t = \begin{cases} \dfrac{r_L e^{(r_B - r_L)t} - r_L}{r_B e^{(r_B - r_L)t} - r_L}, & \text{if } r_B \neq r_L \\ \dfrac{r_B t}{1 + r_B t}. & \text{if } r_B = r_L \end{cases} \tag{5.11}$$

Combining Eq. 5.9 and 5.10, we obtain

$$P(n(t) = x|t) = [1 - P(n(t) = 0|t)](1 - \eta_t)\eta_t^{x-1}, \tag{5.12}$$

for $x \geq 0$.

## 5.2. Forest Representation of $\mathcal{L}$

Instead of representing the reconciliation $\gamma$ as a mapping table from each gene node to a species node, an equivalent, inverse representation of the reconciliation can be formed from the point of view of the species. This representation makes calculating probabilities of reconciliation easier. In this case, we need to know which gene nodes are associated with each branch of the species tree. Given a reconciliation $\gamma$, a species tree branch $(p(s), s) \in E_S$ induces a forest $F(\gamma, s)$. Each tree in the forest represents an embedded subtree of the gene tree that, when mapped to the species tree, resides in branch $(p(s), s)$. In other words, $F(\gamma, s)$ represents the reconciliation restricted to $(p(s), s)$. For each $f \in F(\gamma, s)$, the element $\gamma_f$ is an event-labeled gene subtree with topology $T_f = (V_f, E_f)$ and root $\rho_f$ mapped to $p(s)$. Internal nodes of $\gamma_f$ arose through a gain events; Every internal node is labeled with the event that caused the bifurcation, (i.e. duplication or transfer). The size of $F(\gamma, s)$ is the number of genes that are present in $p(s)$. Each of these genes is the root of a tree in $F(\gamma, s)$. These roots may be inherited by vertical descent from the parent of $p(s)$ or be descendants of a transfer into the species tree branch $(p(p(s)), p(s))$. All leaves of $\gamma_f$ are mapped to species $s$, except for descendants of a transfer event from $(p(s), s)$ to a distant branch. A transfer event will result in a new gene in a species tree branch that is incomparable to the donor species. Transfer edges (and their descendants) are considered part of the embedded gene tree associated with the donor gene until the next speciation. The descendants of a transfer edge that are leaves of $\gamma_f$ will form the roots of trees in the forest of the recipient species.

For example, in Figure 5.1, node $g_6$ is the child of a transfer from $< R, F >$ to $< R, E >$. The gene tree branch $< g_2, g_6 >$ is part of the forest associated with $F$. In addition, $g_6$ is the root of the tree in the forests associated with $C$ and $D$. After a gene transfer, the time for the gene to evolve until the next speciation event may be different from the original species tree branch.

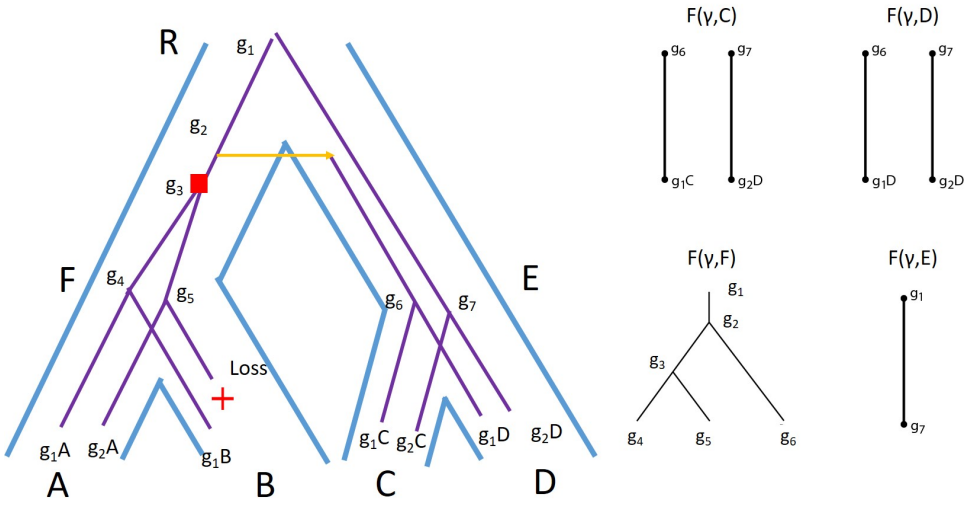



Figure 5.1: Gene tree embedded in species tree. The forests in four species tree branches are shown at right. For each embedded gene tree, the root is a gene that was present at the parent species node. Note that the gene transfer from $g_2$ to $g_6$ in the embedded gene tree in $F(\gamma, F)$ creates a new gene in a distant species tree branch. This new gene is still considered to be a node in the embedded gene tree associated with branch $< R, F >$.

Using the forest representation, the likelihood of the reconciliation can be decomposed into the product of probabilities for all the embedded subtrees, multiplied by a coefficient, $W(\gamma)$, describing number of ways to connect these embedded subtrees:

$$\mathcal{L}^L(\gamma) = W(\gamma) \prod_{s \in V_S \setminus \rho} \prod_{f \in F(\gamma, s)} \mathcal{L}_G(\gamma_f, s). \tag{5.13}$$

$\mathcal{L}_G(\gamma_f, s)$ is the probability that a gene copy from $p(s)$ results in the embedded gene tree $\gamma_f$ when the timeline reaches $s$. $W(\gamma)$ represents the number of ways to obtain the same reconciliation $\gamma$ when connecting the leaves of all gene subtrees embedded in species tree branches, except leaf species, to the roots of gene subtrees embedded in other species tree branches.

Given an embedded gene subtree, there may be several ways to connect it to gene subtrees

in descendant forests in the same reconciliation. (Note that because of transfers, descendant forests may not be in descendant species.) For example, the gene subtree embedded in species tree branch $(\alpha, \beta)$ of Fig 5.2(a) results in the same reconciliation solution as the gene subtree embedded in species tree branch $(\alpha, \beta)$ of Fig 5.2(b). The number of ways to connect topology $T_f$ to subtrees rooted at its leaves is equivalent to the number of ways of reordering the leaves of $T_f$ such that the new tree $T'_f$ represents the same reconciliation as $f$. No known closed-form formula exists to calculate $W(\gamma)$. Instead, this value is calculated using a recursive approach (Arvestad et al., 2009).

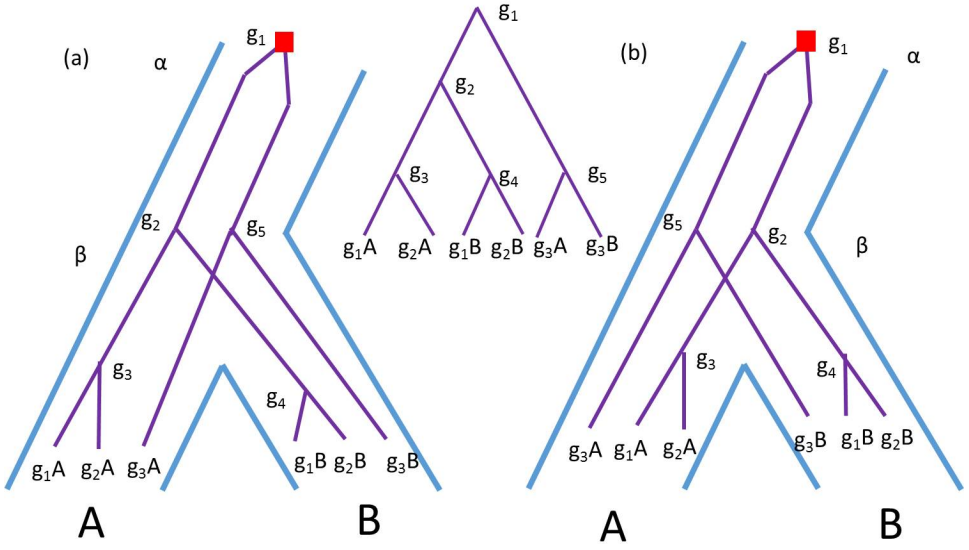



Figure 5.2: Two isomorphic embedded gene trees in the species tree branch $(\alpha, \beta)$ are equivalent and represent the same reconciliation solution.

### 5.2.1. Calculating $\mathcal{L}_G(\gamma, s)$

$\mathcal{L}_G(\gamma_f, s)$ can be calculated by further decomposition:

$$\mathcal{L}_G(\gamma_f, s) = P(\gamma_f | T_f, \vec{r}) P(T_f | k_f) P(k_f | \vec{r}, t_s). \tag{5.14}$$

From right to left, $P(k_f|\vec{r}, t_s)$ is the probability of observing a gene tree with $k_f$ genes, independent of the topology of the tree; $P(T_f|k_f)$ is the probability of observing topology $T_f$, among all possible topologies with $k_f$ leaves; and $P(\gamma_f|T_f, \vec{r})$ is the probability of the given $k_f$ leaves corresponds to a specific series of duplication and transfers.

I explain each term, in detail, in the following three paragraphs. Under different assumptions, different event models may be applied to calculate the third term in the equation, but the first two terms are the same for different scenarios.

**1. Estimating $P(\gamma_f|T_f,\vec{r})$:** In this case, we need to consider the probability of a particular series of duplications and transfers associated with this reconciliation $\gamma_f$. The probability that a node is gained by duplication is $r_D/(r_D+r_T)$, similarly, the probability that a gain event is a transfer is $r_T/(r_D+r_T)$. Therefore, the probability of a particular series of $N_D$ duplications and $N_T$ transfers is:

$$P(\gamma_f|T_f,\vec{r}) \;=\; \frac{r_D^{N_D} r_T^{N_T}}{(r_D+r_T)^{N_D+N_T}}, \tag{5.15}$$

where $N_\epsilon$ represent the number of events of type $\epsilon$ in this embedded gene tree.

**2. Estimating $P(T_f|k_f)$:** The value of $P(T_f|k_f)$ must account for the fact that the given $k_f-1$ gain events may occur in different gene copies resulting in different gene tree topologies. For example, three gain events can result in 6 unlabeled topologies with 4 leaves (Figure 5.3). All of these topologies are equally probable. In addition, given a specific embedded gene tree topology $T_f$, among all the topologies of equal size, some will have the same shape as $T_f$, but with different leaf ordering. These topologies should also be included in the probability. For example, among rooted trees with 4 leaves, 4 topologies have the same shape, and 2 topologies have a different shape (Figure 5.3). The probability of producing a given unlabeled tree topology, $P(T_f|k)$, can not be represented with a closed-form formula. This value can be calculated by an iterative procedure, described by Harding (1971) and reproduced here. To simplify the equation, let $h(T_f)$ represent the probability $P(T_f|k_f)$. Let $T_v$ be a subtree rooted at $v$ and let $\delta(g_1,g_2)$ represent an indicator function such that

$$\delta(g_1,g_2) = \begin{cases} 1, & \text{if} T_{g_1} \text{ and } T_{g_2} \text{are isomorphic,} \\ 0. & \text{otherwise} \end{cases} \tag{5.16}$$

Then $h(T)$ can be calculated using a recursive function:

$$h(T_g) = \frac{2^{\delta(l(g),r(g))}}{|L(T_g)|} h(T_{l(g)})h(T_{r(g)}). \tag{5.17}$$

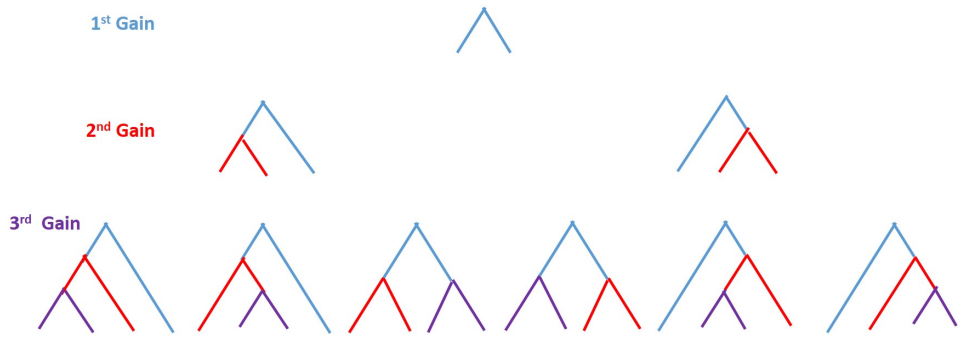



Figure 5.3: All possible rooted trees with four leaves that arose through three gain events. There is only one possible position for the first gain event to occur. Two branches are possible sites for the second gain event to occur, and three for the third gain. In general, the total possible number of unlabeled trees with $k$ leaves produced by $(k-1)$ gain events is $(k-1)!$

**3. Estimating $P(k_f|t_s,\vec{r})$:** The term $P(k|t_s,\vec{r})$ is the probability of observing $k$ gene copies after a time interval, $t_s$. One thing to note is that an embedded gene tree with $k_f$ leaves does not necessarily mean exactly $k_f-1$ gain events occurred; there can be many scenarios that lead to $k_f$ copies. $P(k_f|t_s,\vec{r})$ must account for all combinations of $N_B$ gains and $N_L$ losses such that $1+N_B-N_L=k_f$. The value of this probability can be estimated for the Constant event rate process and for the Birth-Death event process.

First, we discuss how this is estimated for Constant event process. Assuming the event rate is constant throughout the species tree, $N_D$, $N_T$, and $N_L$ all follow a Poisson distribution. Further, as the sum of two Poisson distributed random variables also follows a Poisson distribution, $N_D+N_T$ can also be modelled by a Poisson distribution with mean $(r_D+r_T)t_s$. The distribution of $k_f-1=N_D+N_T-N_L$ is the distribution of the difference between two Poisson distributions. An expression for this distribution, denoted $S(X|\lambda,\mu)$, where $X$ is a random variable with a probability distribution that is equal to the difference between two Poisson distributions, Poi($\lambda$) and Poi($\mu$). This distribution has been derived by Skellam (1946). Thus, the term $P(k_f|t_s,\vec{r})$ for the constant rate process is:

$$P(k_f|t_s,\vec{r}) = S(k_f-1|(r_T+r_D)t_s, r_L t_s). \tag{5.18}$$

Under the Birth-Death process, the term $P(k_f|t_s,\vec{r})$ can be estimated using Eq. 5.12, the modified geometric distribution derived by Kendall (1949):

$$P(k_f|t_s,\vec{r}) = [1-P(0|t_s)](1-\eta_t)\eta_t^{k-1}, \tag{5.19}$$

where $\eta_t = \dfrac{r_L e^{(r_B - r_L)t} - r_L}{r_B e^{(r_B - r_L)t} - r_L}$ when $r_B \neq r_L$.

Combining all three terms, we obtain for the Constant model:

$$\mathcal{L}_G^{CL}(\gamma_f, s) = \frac{r_D^{N_D} r_T^{N_T}}{(r_D + r_T)^{N_D + N_T}} P(T_f|k) S(k-1|(r_T + r_D)t_s, r_L t_s). \tag{5.20}$$

For the Birth-Death model, the likelihood is:

$$\mathcal{L}_G^{BDL}(\gamma_f, s) = \frac{r_D^{N_D} r_T^{N_T}}{(r_D + r_T)^{N_D + N_T}} P(T_f|k)[1 - P(0|t)](1 - \eta_t)\eta_t^{k_f - 1}. \tag{5.21}$$

## 5.3. Event Number Representation of $\gamma$

Sometimes calculating the likelihood of the exact subtree annotated with events is too complicated. An alternative strategy is to use the likelihood of the number of events that occurred in a species branch. The event number approach simply estimates the probability that $N_\epsilon$ events of type $\epsilon$ occurred in a species tree branch,

Under this likelihood representation, the probability of the reconciliation (Eq. 5.4) can be decomposed into the product of the likelihood of the events on each species tree branch:

$$\mathcal{L}^N(\gamma) = \prod_{s \in V_S \setminus \rho} \mathcal{L}_S(\gamma, s), \tag{5.22}$$

where $\mathcal{L}_S(\gamma, s)$ describes the probability of the corresponding gene events in the species tree branch $(p(s), s)$.

This expression is simpler, compared to the likelihood of observing the forest (Eq.5.13), for two reasons. First, this expression of likelihood can be derived directly from the event process, instead of taking the product required in the derivation of $\mathcal{L}_G$. Second, as only the number of events are considered in this likelihood expression, the embedded subtree topology is not considered.

This approach is used in the DrML software package (Górecki and Eulenstein, 2014) for

reconciliation with duplication events only, which seeks a reconciliation that maximizes:

$$\prod_{s \in V_S \setminus \rho} \mathcal{L}_S^N(\gamma, s), \tag{5.23}$$

where

$$\mathcal{L}_S^N(\gamma, s) \;=\; P(N_D | r_D, t_s), \tag{5.24}$$

the probability that $N_D$ duplications occurred in a species tree branch, given the branch length, $t_s$, and duplication event rate $r_D$. This model does not capture the probability of a specific embedded gene tree, but rather captures the probability that $N_D$ gene duplications occurred between two speciations. Only duplications are considered in this likelihood in DrML.

Here, I derive estimates of the likelihood for the DTL event model by calculating the probability that $N_D$ duplications, $N_T$ transfers, and $N_L$ losses occurred on a species tree branch of length $t_s$, given event rates $\vec{r}$:

$$\mathcal{L}_S^N(\gamma, s) \;=\; P(N_D, N_T, N_L | \vec{r}, t_s). \tag{5.25}$$

I derive two different estimates: one for the probability under the Constant Event model, $\mathcal{L}_S^{CN}$, and the other under the Birth-Death process, $\mathcal{L}_S^{BDN}$.

### 5.3.1. Constant Event Process

In the three event model, since all three gene event rates are independent throughout the species tree, the likelihood can be expressed as a product:

$$P(N_D, N_T, N_L | \vec{r}, t_s) = \prod_{\epsilon} P(N_\epsilon | r_\epsilon, t_s). \tag{5.26}$$

Under the assumption that the presence of more gene copies in a genome does not increase the probability of the next gene event, the probability that $N_\epsilon$ events occur, can be modeled by a Poisson distribution:

$$P(N_\epsilon | r_\epsilon, t_s) \;=\; \frac{e^{-r_\epsilon t_s} (r_\epsilon t_s)^{N_\epsilon}}{N_\epsilon!}. \tag{5.27}$$

Substituting Eq.5.27 into Eq. 5.26, we obtain

$$\mathcal{L}_S^{CN}(s) = \frac{e^{-r_D t_s}(r_D t_s)^{N_D}}{N_D!}\frac{e^{-r_T t_s}(r_T t_s)^{N_T}}{N_T!}\frac{e^{-r_L t_s}(r_L t_s)^{N_L}}{N_L!}. \quad (5.28)$$

### 5.3.2. Birth-Death Process

Under the Birth-Death process, as transfers and duplications are both processes that increase the copy number, we consider them both to be birth events. Therefore, the estimation of the probability $P(N_D, N_T, N_L|\vec{r}, t_s)$ can be done in two steps. First, we calculate the probability of $P(N_B, N_L|\vec{r}, t_s)$, where $N_B$ is the sum of $N_T$ and $N_D$. Next, we adjust the probability for different combinations of $N_T$ and $N_D$.

First, I derive the *expected* number of birth and death events over a time interval, $t$, which provides a result that we use later in this section. As the event rate depends on the current gene family size, the expected number of birth $\overline{N}_B(t)$ and death $\overline{N}_L(t)$ events can be estimated using a pair of differential equations:

$$\frac{d\overline{N}_B}{dt} = r_B(1 + \overline{N}_B - \overline{N}_L) \quad (5.29)$$

$$\frac{d\overline{N}_L}{dt} = r_L(1 + \overline{N}_B - \overline{N}_L). \quad (5.30)$$

Solving this coupled differential equation and plugging in the initial conditions ($\overline{N}_B(0) = 0, \overline{N}_L(0) = 0$), we obtain expressions for the expected number of birth and death events at time $t$:

$$\overline{N}_B(t) = \frac{r_b}{r_B - r_L}(\exp[(r_B - r_L)t] - 1), \quad (5.31)$$

$$\overline{N}_L(t) = \frac{1}{r_B - r_L}(\exp[(r_B - r_L)t] - 1). \quad (5.32)$$

Estimating the probability that exactly $N_B$ births and $N_L$ deaths occurred is much more complicated, as I explain here. Let $P_{N_B,N_L}(t)$ denote the probability that $N_B$ birth events and $N_L$ death events occurred during a time interval $t$. Suppose $N_B$ births and $N_L$ deaths occur prior to time $t + \Delta t$, where $\Delta t$ denote a small increment of time. This can occur in three ways:

1. $N_B - 1$ births and $N_L$ deaths occur prior to $t$ and one more birth occurs between $t$ and

$t + \Delta t$;

2. $N_B$ births and $N_L - 1$ deaths occur prior to $t$ and one more death occurs between $t$ and $t + \Delta t$;

3. $N_B$ births and $N_L$ deaths occur prior to $t$ and no events occur between $t$ and $t + \Delta t$.

$P_{N_B,N_L}(t)$ can then be decomposed into a sum of the probabilities of these three scenarios. In each case, the probability of the scenario is the product of the probability that a certain number of events occurred prior to time $t$ and the probability of the event (if any) that occurred between $t$ and $t + \Delta t$. We assume that the probability of more than one event occuring in a small time interval $\Delta t$ is vanishingly small. Note that the probability of an event occurring between $t$ and $t + \Delta t$ depends on $n(t)$, the number of gene copies at time $t$, since each copy can experience an event. Thus, the probability of an event of type $\epsilon$ between $t$ and $t + \Delta t$ is $r_\epsilon \Delta t$. Recall, from Eq. 5.6, $n(t) = N_B(t) - N_L(t) + 1$, assuming there was one gene copy at time $t = 0$.

The probabilities for the three scenarios are:

1. $N_B - 1$ copies were gained and $N_L$ copies were lost prior to time $t$. Thus, the population size at time $t$ is $N_B - N_L$. One gain occurs during $\Delta t$, so the probability is:

$$P_{N_B,N_L}(t + \Delta t) = P_{N_B-1,N_L}(t)(N_B - N_L)r_B \Delta t. \tag{5.33}$$

2. $N_B$ copies were gained and $N_L - 1$ copies were lost prior to time $t$. Thus, the population size at time $t$ is $N_B - N_L + 2$. A loss occurs during $\Delta t$, and the probability is:

$$P_{N_B,N_L}(t + \Delta t) = P_{N_B,N_L-1}(t)(N_B - N_L + 2)r_L \Delta t. \tag{5.34}$$

3. $N_B$ copies were gained and $N_L$ copies were lost prior to time $t$. Thus, the population size at time $t$ is $N_B - N_L + 1$. No event occurs during $\Delta t$. The probability of any event occurring between $t$ and $t + \Delta t$ is $(N_B - N_L + 1)(r_B + r_L)\Delta t$; the probability of no event is thus 1 minus this value. Therefore, the probability is:

$$P_{N_B,N_L}(t + \Delta t) = P_{N_B,N_L}(t)[1 - (N_B - N_L + 1)(r_B + r_L)\Delta t]. \tag{5.35}$$

Combining the probabilities of all three scenarios, we obtain:

$$\begin{aligned} P_{N_B,N_L}(t+\Delta t) =& P_{N_B-1,N_L}(t)(N_B-N_L)r_B\Delta t \\ &+ P_{N_B,N_L-1}(t)(N_B-N_L+2)r_L\Delta t \\ &+ P_{N_B,N_L}(t)[1-(1+N_B-N_L)(r_B+r_L)\Delta]. \end{aligned} \tag{5.36}$$

The above equation becomes a differential equation when $\Delta t \to 0$, yielding:

$$\begin{aligned} \frac{dP_{N_B,N_L}(t)}{dt} =& P_{N_B-1,N_L}(t)(N_B-N_L)r_B \\ &+ P_{N_B,N_L-1}(t)(N_B-N_L+2)r_L \\ &- P_{N_B,N_L}(t)(1+N_B-N_L)(r_B+r_L) \end{aligned} \tag{5.37}$$

One approach to solving this equation is to use Komogorov's Equation to derive the $P_{N_B,N_L}(t)$ theoretically. However, as the number of events, $N_B$ and $N_L$, and the time $t$, are not bounded in this equation, deriving a closed form formula for infinite number of states is intractable.

Instead, we introduce $\phi(N_B, N_L, t)$, a probability generating function (PGF) with two variables for the probability $P_{N_B,N_L}(t)$:

$$\phi(N_B, N_L, t) = \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} N_B^i N_L^j P_{i,j}(t). \tag{5.38}$$

Attempting to obtain a closed-form expression for the function $\phi$, a series of algebraic manipulations (see Appendix 8.4) shows that solving the PGF is equivalent to solving a set of ordinary differential equations (ODE):

$$\frac{dt}{1} = \frac{dN_B}{-N_B\beta} = \frac{dN_L}{-N_L\beta} = \frac{d\phi}{\phi\beta}, \tag{5.39}$$

where $\beta = r_B(N_B-1) - r_L(N_L-1)$. These equations can be reordered and simplified to obtain a second order ODE to solve for $N_B$ (see Appendix 8.4):

$$\frac{d^2N_B}{dt^2} + (2\lambda N_B + \mu - \lambda)\frac{dN_B}{dt} = 0. \tag{5.40}$$

Lacking more restrictions on the problem, a tractable solution of the PGF is hard to find.

Using a third approach, I investigated the possibility that $P_{N_B,N_L}(t)$ could be approximated

using a parametric distribution. As the gene number after time $t$ follows a modified geometric distribution (Eq. 5.12) (Kendall, 1949), the difference between the number of birth events and the number of death events should follow a modified geometric distribution. The waiting times of birth and death events are sampled from the same exponential distribution, and birth and death events occur with the probabilities that are proportional to the ratio between $r_B$ and $r_L$. Therefore, I hypothesize that the distribution that describes $N_B$ and $N_L$ are two similar distributions, and that the difference between those two random variables could be approximated by a geometric distribution. It is important to note that the difference between two geometric distribution does not follow a geometric distribution. This suggests that the number of birth events and the number of death events do not follow geometric distributions. However, as the simulation analysis below shows, the number of births and deaths are often independent and may be approximated by two geometric distributions.

To investigate this approximation, I simulated a Birth-Death process where events occur at a rate $r = r_B + r_L$ over a time interval, $t$. The initial population size is $n(0) = 1$. At each iteration, a waiting time, $t_w$, is sampled from an exponential distribution, $\text{Exp}(rn(t))$ and the elapsed time is incremented by $t_w$. If the elapsed time exceeds a termination threshold, the simulation terminates. Otherwise, a birth (respectively, death) event occurs with probability $r_B/(r_B + r_L)$ (respectively, $r_L/(r_B + r_L)$) and the number of gene copies is incremented (respectively, decremented). The event and the current elapsed time are recorded.

This process was repeated for 100 trials, for various values of $r_B$ and $r_L$ (both from 0.005 to 0.02, incremented by 0.005), with termination time $t$ was set to 100. For each event rate combination, the number of birth and death in 100 gene families was collected and displayed as histogram (red histogram in Figure 5.4, 5.5). The distribution was compared with an estimated distribution, which is drawn as geometric distribution calculated from $p_B, p_D$ using Equations 5.47 and 5.48 (green histogram in Figure 5.4, 5.5). The comparison shows that under the assumptions that the event rates are rare, and if the number of events is a small number on a single species tree branch, the value of $P_{B,D}(t)$ can be approximated by two independent geometric distribution. Here the most number of events for a single species tree branch is 15, which is over the assumption of events are rare for a single species tree branch.

In order to assess how well a geometric distribution approximates $P_{N_B,N_L}(t)$, I compared the birth number and death number from the simulation with the the two geometric distributions with parameters $p_B$ and $p_L$. These parameters were estimated from the expected number of

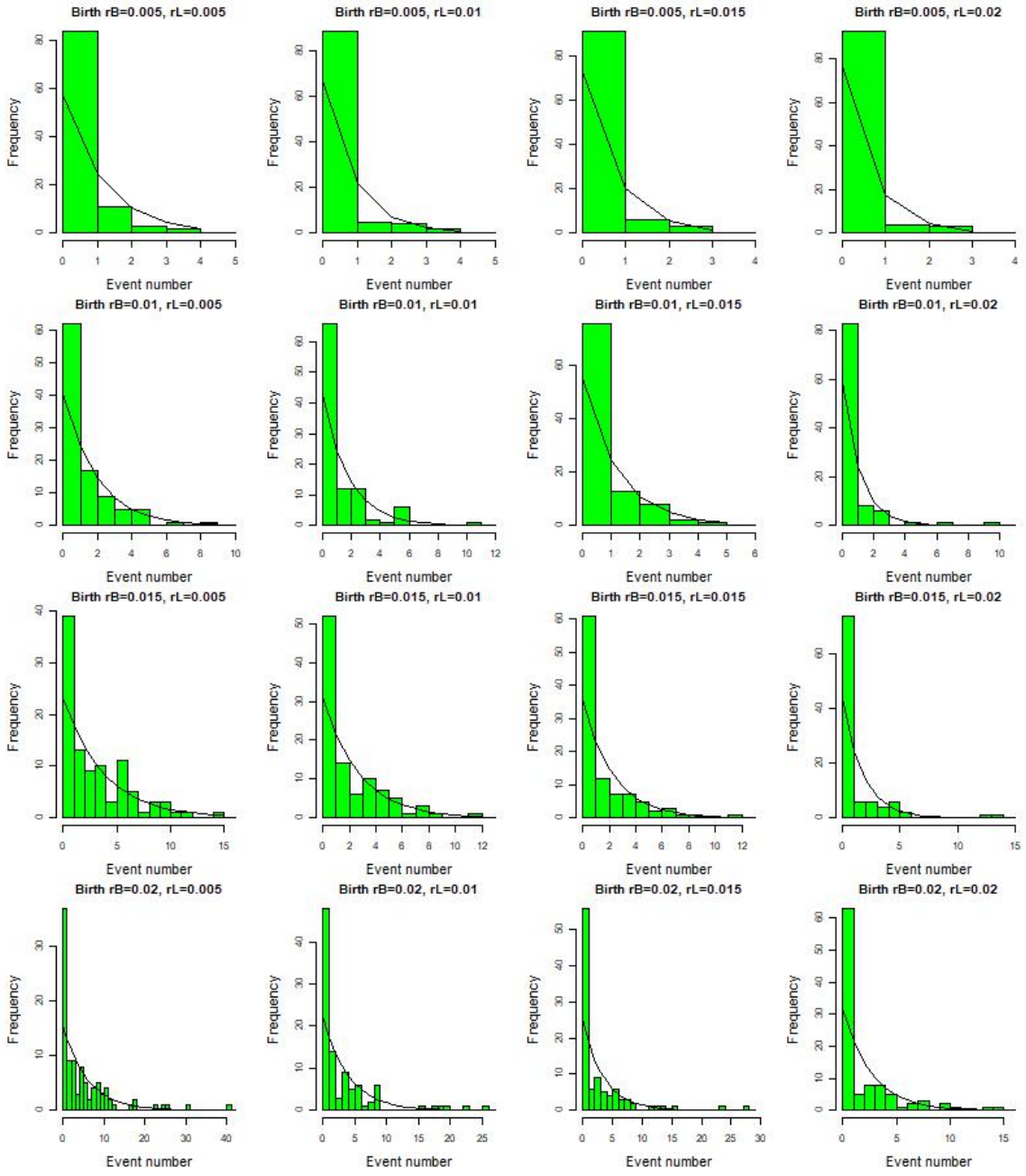



Figure 5.4: Comparison of simulated and estimated number of birth events for all 16 event rate combinations. The simulated number of events are shown in histogram and the line represent the estimated number using standard geometric distribution.

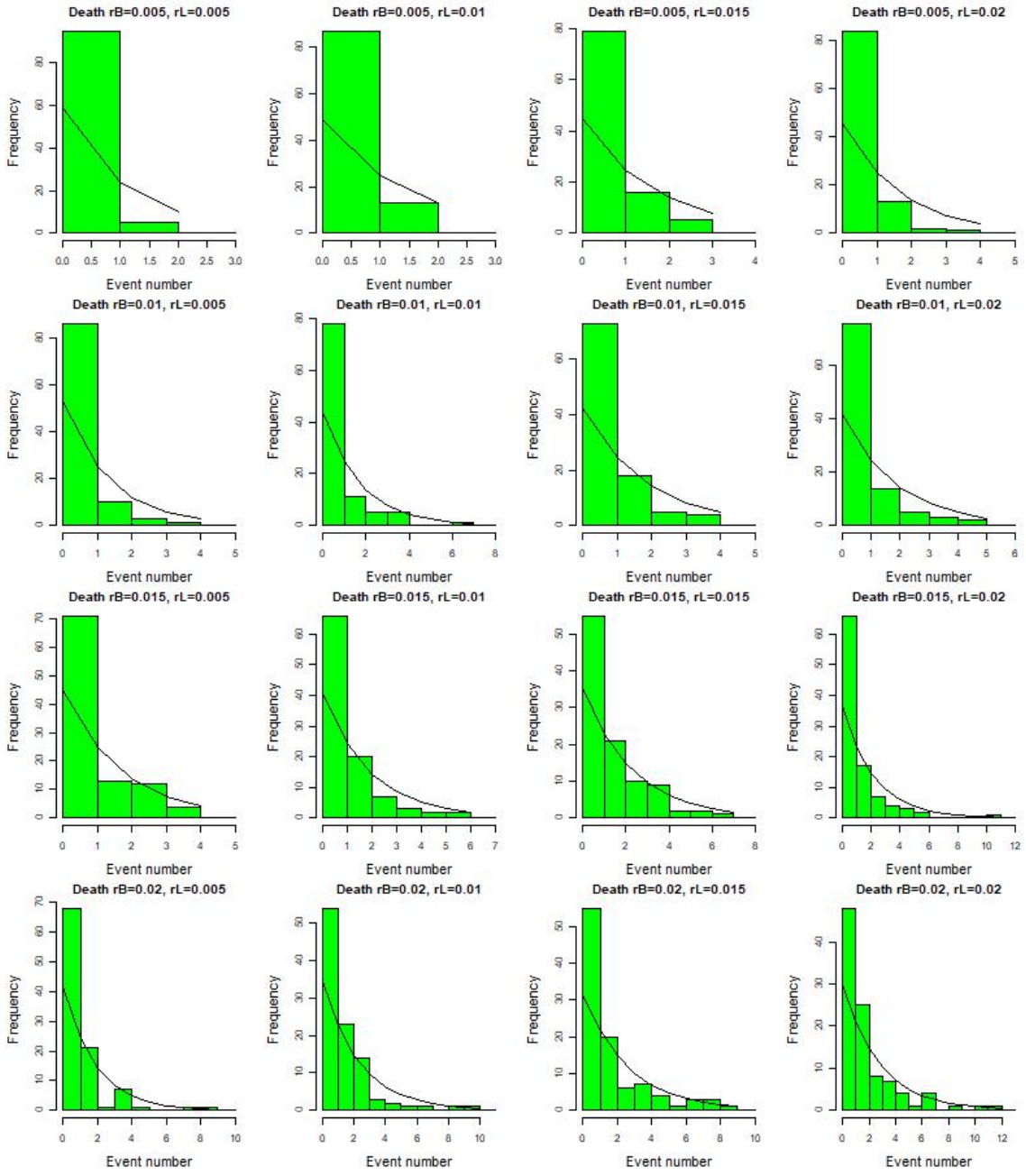



Figure 5.5: Comparison of simulated and estimated number of death events for all 16 event rate combinations. The simulated number of events are shown in histogram and the line represent the estimated number using standard geometric distribution.

birth and death events given in Eq.5.31 and Eq.5.32:

$$p_B = \frac{1}{\overline{N_B}(t)+1} \tag{5.41}$$

$$p_L = \frac{1}{\overline{N_L}(t)+1}. \tag{5.42}$$

The comparison of the theoretical geometric distribution and the histogram of the number of simulated events shows that the distributions have similar shapes. These simulations always exhibit shapes similar to geometric distributions.

The above equations leads to an approximation of $P_{N_B,N_L}(t)$, if we make the further assumption that $N_B$ and $N_L$ are independent:

$$P_{N_B,N_L}(t) \approx P_{N_B}(t)P_{N_L}(t) \tag{5.43}$$

$$\approx (1-p_B)^{N_B}(1-p_L)^{N_L}. \tag{5.44}$$

This probability must be modified to accommodate the specific set of duplications and transfers implied by the reconciliation. (In the event number model, we consider a set, rather than a sequence of duplication and transfer events.) In the case where $N_B$ birth events occurred, the probability that $N_D$ of them are duplications and $N_T = N_B - N_D$ of them are transfers is:

$$\binom{N_B}{N_D}\frac{r_D^{N_D} r_T^{N_T}}{(r_D+r_T)^{N_B}} \tag{5.45}$$

Therefore, the likelihood function can be approximated with:

$$\mathcal{L}_S^{BDN}(s) = (1-p_B)^{N_D+N_T} p_B (1-p_L)^{N_L} p_L \binom{N_D+N_T}{N_T} \frac{r_T^{N_T} r_D^{N_D}}{(r_T+r_D)^{N_T+N_D}}, \tag{5.46}$$

where $p_B$ and $p_L$ are calculated by plugging Eq. 5.31 and Eq. 5.32 into Eq. 5.41 and Eq. 5.42, yielding

$$p_B = \frac{r_B - r_L}{r_B e^{(r_B-r_L)t_s} - r_L} \tag{5.47}$$

$$p_L = \frac{r_B - r_L}{r_B e^{(r_B-r_L)t_s} - 1 + r_B - r_L}. \tag{5.48}$$

## 5.4. Likelihood and Parsimony

Recall that our goal is to consider whether there are conditions under which the most likely reconciliation is also the most parsimonious reconciliation.

In the previous section, I derived expressions for the likelihood of a lineage-induced reconciliation ($\gamma_s$ or $\gamma_f$) under four scenarios ($\mathcal{L}_G^{BDL}, \mathcal{L}_S^{BDN}, \mathcal{L}_G^{CL}, \mathcal{L}_S^{CN}$). Here, for each scenario,

- I first derive an expression for the likelihood for the entire reconciliation, $\mathcal{L}(\gamma)$.
- Next, I obtain a simpler estimate for the log likelihood of $\mathcal{L}(\gamma)$, based on various simplifying assumptions.
- Finally, I compare the log likelihood with the maximum parsimony cost function and ask under what conditions the two could be equivalent.

For all four scenarios, I assume that all species tree branch lengths are of the same, constant length, $\hat{t}$. Additional, scenario-specific assumptions are introduced in the following sections. Our goal is to determine whether it is possible to select costs such that the most parsimonious solution is also the most likely solution. Thus, we seek the costs such that:

$$argmin_{\gamma\in\Gamma} C(N_D, N_T, N_L) = argmax_{\gamma\in\Gamma} \mathcal{L}(\gamma) \tag{5.49}$$

where

$$C(N_D, N_T, N_L) = N_D(\gamma)\cdot C_D + N_T(\gamma)\cdot C_T + N_L(\gamma)\cdot C_L \tag{5.50}$$

In the following sections, I will show that the log likelihood function of the four scenarios are all of a generic form:

$$\mathcal{L} = \left( \sum_{\epsilon\in\{T,D,L\}} N_\epsilon \cdot f_\epsilon(\vec{r}, \hat{t}) \right) + g(N_D, N_T, N_L) + K, \tag{5.51}$$

where $K$ is a constant that only depends on $\vec{r}$ and $\hat{t}$, but not on the reconciliation $\gamma$. This constant can be neglected in the optimization criteria as it is the same for all solutions. Further if $C_\epsilon = -f_\epsilon(\vec{r}, \hat{t})$, and $g(N_D, N_T, N_L)$ can be ignored, then the two optimization criteria are equivalent. We then consider the conditions under which $g$ can be ignored for the four scenarios.

### Calculating Lineage Log Likelihood ($\mathcal{L}^L$)

Recall that in the lineage model, the likelihood of reconciliation $\gamma$ is:

$$\mathcal{L}^L(\gamma) = W(\gamma) \prod_{s \in V_S \setminus \rho} \prod_{f \in F(\gamma,s)} \mathcal{L}_G(\gamma_f, s). \tag{5.52}$$

Substituting the expression for $\mathcal{L}_G(\gamma_f, s)$ given in Eqn. 5.14 yields:

$$\mathcal{L}^L(\gamma) = W(\gamma) \prod_{s} \prod_{f \in F(\gamma,s)} P(T_f|k) P(k|\vec{r}, t_s) P(\gamma_f|T_f) \tag{5.53}$$

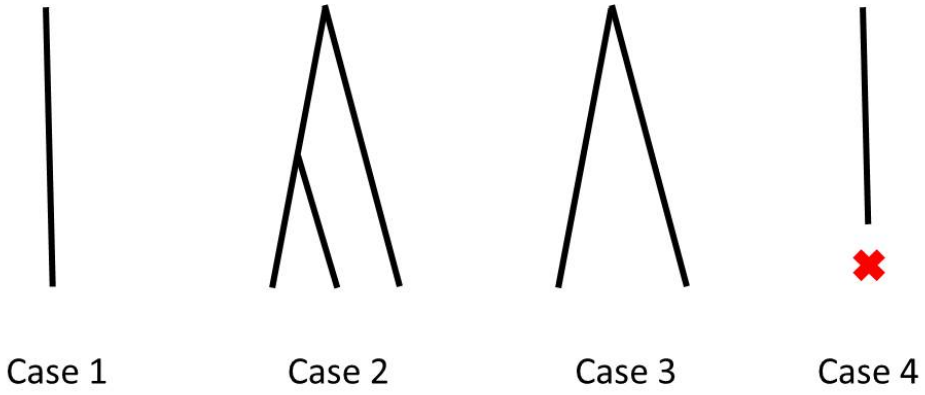



Figure 5.6: The four cases based on the assumption that the embedded gene subtrees have at most 3 leaves.

Under the assumption that the event rates are low, we further assume that only the embedded gene subtrees with at most 3 leaves have a likelihood that is substantially greater than zero. This suggests that there are four possible family sizes ($k_f \in \{0, 1, 2, 3\}$) for any embedded gene subtree in a forest. In Figure 5.6, these correspond to (case 1) a gene tree with a single branch ending at the speciation; (case 2) a gene tree with a single gain, resulting in two leaves; (case 3) a gene tree with two gains, resulting in three leaves; and (case 4) a single branch ending in a loss. For each of these, there is only one tree shape. Therefore, $P(T_f|k_f) = 1$. Assuming the value of $W(\gamma)$ is kept constant among different solutions, and therefore may be negligible, the likelihood function is simplified to the following form:

$$\mathcal{L}^L(\gamma) = \prod_{s} \prod_{f \in F(\gamma,s)} P(k|\vec{r}, t_s) P(\gamma_f|T_f) \tag{5.54}$$

**Birth-Death Model** :

Combining Eq. 5.54 with the approximation for the Birth-Death Model given in Eq. 5.21, we obtain:

$$\mathcal{L}^{BDL}(\gamma) = \prod_{s} \prod_{f \in F(\gamma,s)} [1 - P(0|\vec{r}, t_s)](1 - \eta_t)\eta_t^{k_f - 1} \frac{r_D^{N_D(f)} r_T^{N_T(f)}}{(r_D + r_T)^{N_D(f)+N_T(f)}}, \tag{5.55}$$

where $N_\epsilon(f)$ is the number of events of type $\epsilon$ in the embedded subtree, $\gamma_f$. $P(0|\vec{r}, t_s)$ and $\eta_t$ are defined in Eq. 5.9 and Eq. 5.11; both quantities depend only on $\vec{r}$ and $t_s$. Given that $P(0|\vec{r}, t_s)$ and $\eta_t$ are independent of $f$ and $s$, under the assumption that $t_s = \hat{t}$, the first term,

$$\prod_{s} \prod_{f \in F(\gamma,s)} [1 - P(0|\vec{r}, t_s)](1 - \eta_t)\eta_t^{k_f - 1}, \tag{5.56}$$

can be simplified and the product can be expressed as an exponent:

$$[1 - P(0|\vec{r}, \hat{t})]^{\sum_s |F(\gamma,s)|}(1 - \eta_t)^{\sum_s |F(\gamma,s)|}\eta_t^{\sum_s \sum_f (k_f - 1)}. \tag{5.57}$$

The exponents in this term, $\sum_s |F(\gamma, s)|$ and $\sum_s \sum_f (k_f - 1)$, represent the total number of birth events and the total number of gene subtrees in all species branches, respectfully. The total number of birth events is easy to calculate:

$$\sum_s \sum_l (k_f - 1) = N_D(\gamma) + N_T(\gamma). \tag{5.58}$$

The induced forests for a given reconciliation consist of embedded trees wherein the roots are bifurcations resulting from speciation events (i.e., "speciation nodes") and internal nodes arose via duplications or transfers. Every speciation node is the root of two embedded gene trees, one in each of forest associated with the two branches descending from that species. Therefore, the number of embedded trees is twice the number of speciation nodes in the reconciled gene tree (e.g. Figure 5.7). Since every gene loss adds an internal node in the reconciled tree, the total number of internal nodes in $\gamma$ is $(|V_G| - 1)/2 + N_L(\gamma)$. Since every internal node in $T_G$ arose via speciation, duplication or transfer, the number of speciation nodes is equal to the number of internal nodes in the reconciled gene tree less the number of duplication and transfer events associated with $\gamma$. The total number of internal gene nodes is calculated as:

$$N_{internal} = \frac{|V_G| - 1 + 2N_L}{2} \tag{5.59}$$

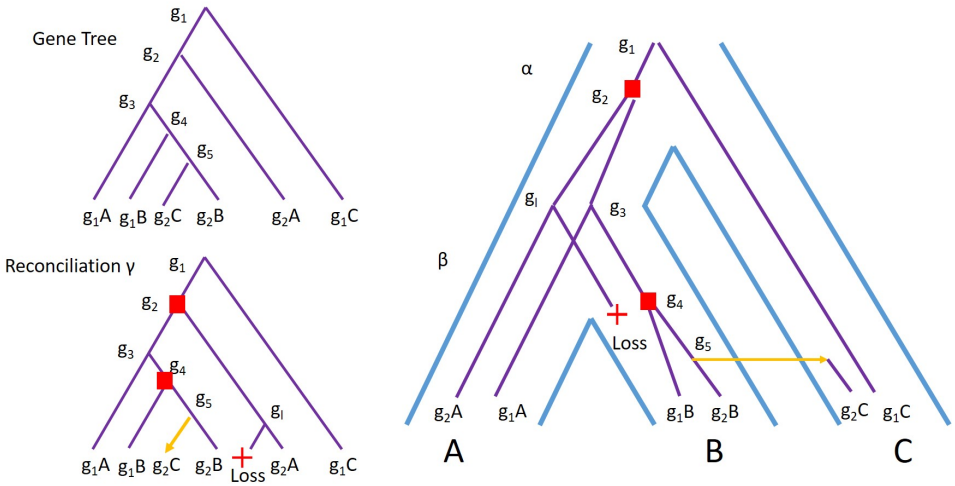



Figure 5.7: The gene tree, reconciliation $\gamma$, and the gene tree embedded in the species tree. The total number of nodes in the reconciliation $\gamma$ is no longer $|V_G|$, which is the number of nodes of the gene tree. The total number of gene nodes in a reconciliation is: $|V_G| + 2N_L$. In this reconciliation, the "speciation nodes" are: $g_1$, $g_l$, and $g_3$, which are roots of two embedded gene trees. $g_2, g_4$ arose through gene duplication event, and $g_5$ has a transfer event. The total size of the reconciliation is 13, which is calculated from $|V_G| + 2N_L$. The number of forests in the embedded species tree is 6. This is consistent with the conclusion that the number of forests is twice the number of "speciation nodes".

and as the internal nodes either encountered a gain event or a speciation event. The number of "normal" internal nodes is:

$$N_{normal} = N_{internal} - N_D - N_T \tag{5.60}$$

$$= \frac{|V_G| - 1 + 2N_L}{2} - N_D - N_T \tag{5.61}$$

Because every "normal" gene node creates towo embedded subtree via speciation event, the total number of gene subtrees is:

$$\sum_s |F(\gamma, s)| = |V_G| + 2N_L(\gamma) - 2N_D(\gamma) - 2N_T(\gamma) - 1. \tag{5.62}$$

Plugging Eq. 5.58 and Eq. 5.62 into Eq. 5.57, the entire first term of Eq. 5.55 becomes:

$$\left[[1 - P(0|\vec{r}, \hat{t})](1 - \eta_t)\right]^{|V_G|+2N_L(\gamma)-2N_D(\gamma)-2N_T(\gamma)-1} \eta_t^{N_D(\gamma)+N_T(\gamma)} \tag{5.63}$$

The second term in Eq.5.55,

$$\prod_{s} \prod_{f \in F(\gamma,s)} \frac{r_D^{N_D(f)} r_T^{N_T(f)}}{(r_D + r_T)^{N_D(f)+N_T(f)}} \tag{5.64}$$

can be simplified by replacing a product with a sum in the exponent, yielding

$$\frac{r_D^{\sum_s \sum_f N_D(f)} r_T^{\sum_s \sum_f N_T(f)}}{(r_D + r_T)^{\sum_s \sum_f (N_D(f)+N_T(f))}}. \tag{5.65}$$

We further simplify by observing that

$$N_\epsilon(\gamma) = \sum_{s \in V_S} N_\epsilon(s) = \sum_{s \in V_S} \sum_{f \in F(\gamma,s)} N_\epsilon(f). \tag{5.66}$$

Substitution the left hand side for the sums in Eq.5.65, we obtain

$$\frac{r_D^{N_D(\gamma)} r_T^{N_T(\gamma)}}{(r_D + r_T)^{N_D(\gamma)+N_T(\gamma)}}. \tag{5.67}$$

Putting Eq.5.63 and Eq.5.67 together, we obtain:

$$\mathcal{L}^{BDL} = \big[[1 - P(0|\vec{r}, \hat{t})](1 - \eta_t)\big]^{|V_G|+2N_L(\gamma)-2N_D(\gamma)-2N_T(\gamma)} \eta_t^{N_D(\gamma)+N_T(\gamma)} \frac{r_D^{N_D(\gamma)} r_T^{N_T(\gamma)}}{(r_D + r_T)^{N_D(\gamma)+N_T(\gamma)}}. \tag{5.68}$$

Taking a log on both side yields the log likelihood:

$$\begin{aligned} \log(\mathcal{L}^{BDL}) = {} & (|V_G| + 2N_L(\gamma) - 2N_D(\gamma) - 2N_T(\gamma)) \log([1 - P(0|\vec{r}, \hat{t})](1 - \eta_t)) \\ & + (N_D(\gamma) + N_T(\gamma)) \log(\eta_t) + N_D(\gamma) \times (\log(r_D) - log(r_D + r_T)) \\ & + N_T(\gamma) \times (\log(r_T) - \log(r_D + r_T)). \end{aligned} \tag{5.69}$$

Assembling terms associated with each event type, we obtain

$$\begin{aligned} \log(\mathcal{L}^{BDL}) = {} & N_D(\gamma) \times (\log(\eta_t) + \log(r_D) - log(r_D + r_T) - 2\log([1 - P(0|\vec{r}, \hat{t})](1 - \eta_t))) \\ & + N_T(\gamma) \times (\log(\eta_t) + \log(r_T) - log(r_D + r_T) - 2\log([1 - P(0|\vec{r}, t_s)](1 - \eta_t)) \\ & + N_L(\gamma) \times (2\log([1 - P(0|\vec{r}, t_s)](1 - \eta_t))) + K, \end{aligned} \tag{5.70}$$

where $K$ is a constant term that does not depend on $N_\epsilon$.

Conveniently, this likelihood function does not contain an interaction term that depends on multiple event types. In fact, the weight for each event is very straightforward to calculate:

$$C_D = -(\log(\eta_t) + \log(r_D) - log(r_D + r_T) - 2\log([1 - p_0(\hat{t})](1 - \eta_t))) \quad (5.71)$$

$$C_T = -(\log(\eta_t) + \log(r_T) - log(r_D + r_T) - 2\log([1 - p_0(\hat{t})](1 - \eta_t))) \quad (5.72)$$

$$C_L = -(2\log([1 - p_0(\hat{t})](1 - \eta_t))) \quad (5.73)$$

When $C_\epsilon$ are set according to the event rates, and the assumption that events are rare implies that no embedded subtree has more than 3 leaves, then the solution that maximum likelihood also has the minimal cost.

**Constant Event Model** :

The probability of observing $k_f$ gene copies is equivalent to the probability of observing a combination of gains and losses, such that $N_B - N_L = k_f - 1$. Here we assume that event rates are sufficiently low that no embedded gene tree in $F(\gamma, s)$ contains both gain and loss events. Each forest $F(\gamma, s)$ can be partitioned into two types: $F(\gamma, s)^+$ for cases 1, 2, 3 and $F(\gamma, s)^-$ for case 4.

The inner product in Equation 5.54 can be split into two products:

$$\mathcal{L}^{CL}(\gamma) = \left( \prod_{s \in V_S \setminus \rho} \prod_{f \in F(\gamma,s)^+} P(k|\vec{r}, \hat{t}) P(\gamma_f | T_f) \right) \left( \prod_{s \in V_S \setminus \rho} \prod_{f \in F(\gamma,s)^-} P(k|\vec{r}, \hat{t}) P(\gamma_f | T_f) \right), \quad (5.74)$$

allowing us to use separate expressions for the probability of the case where $N_L = 0$ and $N_B \geq 0$ and the case where $N_B = 0$ and $N_L > 0$.

In Equation 5.20, we obtained an expression for $P(k|\vec{r}, \hat{t})$ using the Skellum distribution, which is the probability distribution of the net number of gains and losses modeled as the difference between two Poisson distributions. This expression accounts for all possible combinations of births and deaths, including event histories that are not parsimonious. Under the restricted assumption that gain and loss events do not occur together in an embedded subtree, a Skellum distribution is not required. Therefore, this probability can be estimated by Poisson distribution.

In the first term, the probability $P(k_f|\vec{r}, \hat{t})$ represents the probability of observing $N_D + N_T$

gain events, which can be modeled by a Poisson distribution with rate $(r_D + r_T)\hat{t}$.

$$P(k|\vec{r},\hat{t}) = \frac{e^{-(r_D+r_T)\hat{t}}((r_D+r_T)\hat{t})^{(N_D(f)+N_T(f))}}{(N_D(f)+N_T(f))!}. \tag{5.75}$$

Applying this approximation to the first term in Equation 5.74 and plugging in Equation 5.15 results in the first term:

$$\prod_{s\in V_S\setminus\rho}\prod_{f\in F(\gamma,s)^+}\frac{e^{-(r_D+r_T)\hat{t}}((r_D+r_T)\hat{t})^{(N_D(f)+N_T(f))}}{(N_D(f)+N_T(f))!}\frac{r_d^{N_D(f)}r_T^{N_T(f)}}{(r_d+r_T)^{N_D(f)+N_T(f)}}. \tag{5.76}$$

This term can be merged by canceling out the term $(r_d + r_T)^{N_D(f)+N_T(f)}$, yielding:

$$\prod_{s}\prod_{f\in F(\gamma,s)^+}\frac{e^{-(r_D+r_T)\hat{t}}(r_D t_s)^{N_D(f)}(r_T\hat{t})^{N_T(f)}}{(N_D(f)+N_T(f))!}. \tag{5.77}$$

Plugging in Equation 5.66, the entire first term becomes:

$$\frac{e^{-(r_D+r_T)\hat{t}}(r_D\hat{t})^{N_D(\gamma)}(r_T\hat{t})^{N_T(\gamma)}}{\prod_s\prod_{f\in F(\gamma,s)}(N_D(f)+N_T(f))!}. \tag{5.78}$$

When $\gamma_f \in F(\gamma,s)^-$, $P(k_f|\vec{r},\hat{t})$ is the probability that $N_L(f) \geq 1$ or $1 - e^{-r_L\hat{t}}$. And because there is no gain events in these lineages, the value of $P(\gamma_f|T_f)$ is always 1.

Therefore, the second term of Equation 5.74,

$$\prod_{f\in F(\gamma,s)^-}P(k|\vec{r},\hat{t})P(\gamma_f|T_f), \tag{5.79}$$

can be simplified to

$$(1-e^{-r_L\hat{t}})^{N_L(\gamma)}. \tag{5.80}$$

Combining the first and second term, the lineage likelihood function for the reconciliation under Constant Event Model is:

$$\mathcal{L}^{CL} = (1-e^{-r_L\hat{t}})^{N_L(\gamma)}\frac{e^{-(r_D+r_T)\hat{t}}(r_D\hat{t})^{N_D(\gamma)}(r_T\hat{t})^{N_T(\gamma)}}{\prod_s\prod_{f\in F(\gamma,s)}(N_D(f)+N_T(f))!}. \tag{5.81}$$

The log likelihood is easily calculated:

$$\begin{aligned}\log(\mathcal{L}^{CL}) = & N_D(\gamma) \times \log(r_D\hat{t}) \\ & + N_T(\gamma) \times \log(r_T\hat{t}) \\ & + N_L(\gamma) \times \log(1 - e^{-r_L\hat{t}}) \\ & - \sum_s \sum_{f \in F(\gamma,s)} \log((N_D(f) + N_T(f))!) + K\end{aligned} \tag{5.82}$$

The term $\log((N_D(f) + N_T(f))!)$ is an increasing function with positive acceleration as $N_D(f) + N_T(f)$ increases (Fig 5.8a).

This suggests that for a fixed number of gains ($N_B = N_D + N_T$), the value of

$$\sum_s \sum_{f \in F(\gamma,s)} \log((N_B(f)!)$$

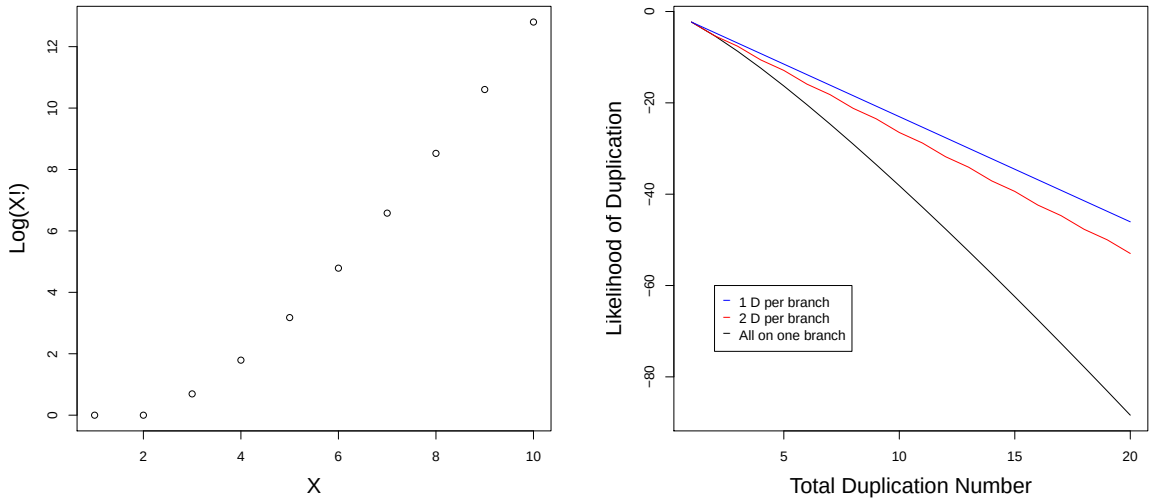
is minimized when the variance of all gain events is minimized. In other words, if the $N_B$ gain events are evenly distributed among all species tree branches, then this value would be substantially smaller. For example, given a species tree with 10 branches, if there is one duplication per branch, the value would be $10 * \log(1) = 0$, compare to 10 duplications on one branch: $\log(10!) = 15$. Therefore, given a species tree with $k$ branches, and mean number of events per branch $r_B\hat{t}$ is small, if we assume the event rate is small enough that the probability to have many event occurring in one branch is sufficiently small, then the term $\sum_s \sum f \in F(\gamma, s) \log(N_B(f)!)$ can be neglected.

When deriving an expression for $\mathcal{L}^{CL}$ in the beginning of this section, we introduced the simplifying assumption that the embedded gene subtrees are only size 3 or less, then the number of duplications and transfers, combined, cannot exceed 2. This leads to this term to be small and may also be neglected. If we further assume that the $r_B$ is low enough so that the probability of more than one gain event per embedded gene tree is negligible, the term $\log(N_B!) = \log(0!) = \log(1!) = 0$. Then the entire term can be ignored in the final likelihood function.

Under this assumption, the cost of event type $\epsilon$ can be set to $-log(r_\epsilon\hat{t})$ so that the solution

that maximizes the likelihood also minimizes the cost.

$$C_D = \log(r_D \hat{t}) \tag{5.83}$$
$$C_T = \log(r_T \hat{t}) \tag{5.84}$$
$$C_L = \log(1 - e^{-r_L \hat{t}}) \tag{5.85}$$




(a) $\log(x!)$ (b) $N_D \log(r_D) - \sum_s \log(x!)$

Figure 5.8: (a) The $\log(x!)$ function. (b) The likelihood value as a function of gain event. Blue represents the condition where each branch may have 0 or 1 gain event; red represents the condition where each branch may have 0 or 2 gain events; black represents the condition where all gain occur in one species tree branch.

### Calculating Log Likelihood for Event Number Representation($\mathcal{L}^N$)

The likelihood function for event numbers simply estimate the probability of $N_T$ transfers, $N_D$ duplications, and $N_L$ losses associated with reconciliation $\gamma$.

### Birth-Death Model :

Substituting the expression for $\mathcal{L}_S^{BDN}$ in Eqn. 5.22 with Eqn. 5.46 result in:

$$\mathcal{L}^{BDN} = \prod_s (1-p_B)^{N_D(s)+N_T(s)} p_B (1-p_L)^{N_L(s)} p_L \binom{N_D(s)+N_T(s)}{N_T(s)} \frac{r_T^{N_T(s)} r_d^{N_D(s)}}{(r_T+r_D)^{N_T(s)+N_D(s)}}, \tag{5.86}$$

where

$$p_B = \left(\frac{r_b}{r_B - r_L}(\exp\left[(r_B - r_L)t\right] - 1) + 1\right)^{-1} \tag{5.87}$$

$$p_L = \left(\frac{1}{r_B - r_L}(\exp\left[(r_B - r_L)t\right] - 1) + 1\right)^{-1}. \tag{5.88}$$

When $t_s = \hat{t}$, $p_L$ and $p_B$ are independent of $s$. Therefore, this equation simplifies to: $\mathcal{L}^{BDN} =$

$$(1-p_B)^{N_D(\gamma)+N_T(\gamma)} p_B (1-p_L)^{N_L(\gamma)} p_L \frac{r_T^{N_T(\gamma)} r_d^{N_D(\gamma)}}{(r_T+r_d)^{N_T(\gamma)+N_D(\gamma)}} \prod_s \binom{N_D(s)+N_T(s)}{N_T(s)}, \tag{5.89}$$

The log likelihood is:

$$\begin{aligned} \log(\mathcal{L}^{BDN}) = {} & N_D(\gamma) \times (\log(\frac{r_D}{r_T+r_D}) + \log(1-p_B)) \\ & + N_T(\gamma) \times (\log(1-p_B) + \log(\frac{r_T}{r_T+r_D})) \\ & + N_L(\gamma) \times \log(1-p_L) \\ & + \sum_s \log \binom{N_D(s)+N_T(s)}{N_T(s)} + K, \end{aligned} \tag{5.90}$$

The term

$$g(N_D, N_T, N_L) = \sum_s \log \binom{N_D(s)+N_T(s)}{N_T(s)} \tag{5.91}$$

may be ignored when the binomial coefficient is equal to a constant on all species tree

branches. This will occur whenever $N_D(s) = 0$ or $N_T(s) = 0$. This assumption is a reasonable assumption as lateral gene transfer rate in prokayotes is estimated to be a major force of genetic variation (Treangen and Rocha, 2011) and gene duplication may occur more often in eukaryotes, suggesting that in different sections of the species tree, the species tree branches may be dominated by either transfers or duplications, but not both. In this case, when

$$C_D = -\log(\frac{r_D}{r_T + r_D}) - \log(1 - p_B) \tag{5.92}$$

$$C_T = -\log(1 - p_B) - \log(\frac{r_T}{r_T + r_D}) \tag{5.93}$$

$$C_L = -\log(1 - p_L), \tag{5.94}$$

the solution that maximizes the likelihood also minimizes the cost.

**Constant Event Model** :

Substituting $\mathcal{L}_S^{CN}$ in Eqn. 5.22 with Eqn. 5.28 results in:

$$\mathcal{L}^{CN} = \prod_s \frac{e^{-r_D\hat{t}}(r_D\hat{t})^{N_D(s)}}{N_D(s)!} \frac{e^{-r_T\hat{t}}(r_T\hat{t})^{N_T(s)}}{N_T(s)!} \frac{e^{-r_L\hat{t}}(r_L\hat{t})^{N_L(s)}}{N_L(s)!} \tag{5.95}$$

$$= \frac{e^{-r_D\hat{t}}(r_D\hat{t})^{N_D(\gamma)} e^{-r_T\hat{t}}(r_T\hat{t})^{N_T(\gamma)} e^{-r_L\hat{t}}(r_L\hat{t})^{N_L(\gamma)}}{\prod_s N_D(s)!N_T(s)!N_L(s)!} \tag{5.96}$$

Calculating the log likelihood yields:

$$\begin{aligned} \log(\mathcal{L}^{CN}) = & N_D(\gamma) \times \log(r_D\hat{t}) \\ & + N_T(\gamma) \times \log(r_T\hat{t}) \\ & + N_L(\gamma) \times \log(r_L\hat{t}) \\ & - \sum_s log(N_D(s)!) - \sum_s log(N_T(s)!) - \sum_s log(N_L(s)!) + K, \end{aligned} \tag{5.97}$$

Since we assume that all events are independent and behave in the same way, it is sufficient to analyze one event type. Here I use duplication as an example. The associated likelihood of duplication is $N_D \log(r_D\hat{t}) - \sum_s \log(N_D(s)!)$, where $r_D\hat{t}$ represent the mean number of duplication in a single species tree branch, $N_D$ represents the total number of duplications and $N_D(s)$ represents the number of duplications that occurred in species tree branch $s$. This term also contains a $log(x!)$ term, which was already discussed in constant model for

the forest representation. Similarly, if the duplication events distributed in the species tree branch evenly, the value of $\sum_s \log(N_D(s)!)$ would be quite small.

As an example, we consider a species tree with 100 branches where $r_B\hat{t} = 0.1$ and the number of gains range from 1 to 20. We can calculate the value of the likelihood of gain for the following three cases and display the result in Figure 5.8b: 1) each branch may have 0 or 1 gain event (blue line), 2) each species tree branch may have either 0 or 2 gain events (red line), 3) all gain events occur in a single species tree branch (black line). This suggests that with low average number of events per branch, and evenly distributed duplication event in the species tree, the value of $N_D \log(r_D\hat{t}) - \sum_s \log(N_D(s)!)$ can be linearly dependent on the number of duplication events. The coefficient of the linear function can be directly calculated: $\log(r_D\hat{t})$.

Under these assumptions, the maximum likelihood and maximum parsimony optimization criteria are equivalent when

$$C_D = -\log(r_D\hat{t}) \tag{5.98}$$

$$C_T = -\log(r_T\hat{t}) \tag{5.99}$$

$$C_L = -\log(r_L\hat{t}). \tag{5.100}$$

As the MP optimization criteria assumes that the least cost to explain events is the best solution. It was a popular belief that the this optimization criteria is appropriate for situations where event rates are low. This study, I derived likelihood functions for reconciliation solutions under different assumptions of how events occurs and suggested the conditions in which these likelihood are appropriate. The comparisons of these likelihood functions and the MP criteria, provide more concrete evidence on the range of event rates that are considered "low". This would facilitate the users when they apply reconciliation software in real data sets and assess if the results are trustworthy.

However, in many cases, these assumptions may not be appropriate. For example, the species tree branches may not be the same length; the gene tree may be much larger than species tree, indicating rapid expansion on some branches, which suggests that some embedded subtree may have more than 4 leaves; or it is hard to estimate the event rates from limited data set. In these cases, the entire reconciliation result obtained by MP model may not have strong statistical support, but may still contain valuable information on some part of the gene tree based on user's intuition. The next section describes a guideline for choosing cost

for reconciliation based on user intuition and species tree.

## 5.5. Conclusion

In this chapter, I started out with the overarching question: Under what conditions are the same results obtained with a parsimony model and a likelihood model? This question arose early in the history of molecular phylogenetics when Farris (1973) showed that under some reasonable assumptions, the reconstructed gene trees under ML and MP are equivalent. However, Felsenstein (1973) later presented counter examples and suggested that Farris' assumptions are "not so reasonable".

Here I compared maximum parsimony and maximum likelihood models in the context of phylogenetic reconciliation, focusing on the problem of how to choose the best set of event costs. Specifically, if the event rates are known, can suitable costs be estimated from the rates? Since the maximum parsimony optimization criterion is a linear function of the number of inferred events, I ask a simpler question: Can the log likelihood be represented as a linear function of the number of events of each type? In this chapter, I considered two likelihood functions: the probability of observing a given number of gene copies in each species tree node, denoted $\mathcal{L}^N$, and the probability of observing a particular reconciliation of the gene tree in the species tree, denoted $\mathcal{L}^L$. For each of these, the probability was calculated using two event models: (1) a birth-death model, in which the probability of an event depends on gene family size, and (2) a constant event model, in which the probability of an event is constant, independent of the number of gene copies.

I first derive general expressions for the likelihoods under these four models under some simplifying assumptions. These general expressions could also be used for other applications, such as software implementations to calculate likelihood of reconciliations. I then derived simple expressions for the log likelihoods of the four models, making further simplifying assumptions as needed, and compared the resulting expressions with the linear cost function under the MP model.

Two assumptions were applied to all four models:

1. The branch length is the same for all branches in the species tree. For all four models, the overall likelihood is a product of the likelihoods associated with each branch in

the species tree. This assumption allows this product to be reduced to a closed form expression.

2. The so-called "ghost" gain events, described in Arvestad et al. (2009), do not occur. The models I use account for gain events on the same species branch that cannot be observed (i.e., gain followed by immediate loss). However, they do not account for gains that are inherited by descendant species and subsequently deleted by parallel losses in all of those descendants.

Several additional assumptions were made specifically for specific models. For the lineage likelihood models, two quantities cannot be obtained as a close form solution:

1. The probability of observing a given reconciliation, $P(T|k_f)$, must account for the number of isomorphic topologies with $k_f$ leaves. To avoid calculating this quantity, I make the additional assumption that the number of leaves in any embedded subtree on a single species tree branch is less than four. For any rooted tree with at most three leaves, there is only one topology, making it possible to ignore this term. Note that a recursive function for calculating this value, derived by Harding (1971), can be used in contexts where a closed form is not needed.

2. It is also necessary to account for the number of ways the leaves of the embedded gene subtrees in one species tree branch can be connected to the roots of embedded gene subtrees in the species branches that descend from it. I make the further simplifying assumption that this quantity, denoted $W(\gamma)$, is a constant for all reconciliations of a given gene tree and a species tree. Since we are interested in the reconciliation that maximizes the likelihood, rather than actual value of the likelihood, a constant term can be ignored.

   The error associated with this assumption will depend on how much $W(\gamma)$ varies over those reconciliations that have non-negligible probabilities. Every reconciliation must entail a set of gain and loss events that results in $L_N(T_G)$ leaves. This constrains how much $W(\gamma)$ can vary. The variance in $W(\gamma)$ over the set of possible reconciliations will depend on the distribution of these events over the species tree branches. Intuitively, $W(\gamma)$ will tend to be greater when all gains occur on one species tree branch and smaller when they are evenly distributed. Exploring how much $W(\gamma)$ varies in practice is an interesting direction for future work, as is deriving bounds on the variation of W().

For the lineage likelihood with constant event rate ($\mathcal{L}^{CL}$), calculating the likelihood involves a Skellum distribution, which is not closed form. Therefore, I also assume that the scenario where a gain event occurred and immediately lost does not occur in gene tree lineages. This allows the Skellum distribution to be replaced with a Poisson distribution (Eqn. 5.75).

For the event number likelihood functions with the birth-death event model, I further assume that the probability of the number of events can be estimated with a geometric distribution (Eqn. 5.75). Note that most of these assumptions are most likely to be justified when the event rates are small, which is appropriate for a maximum parsimony model. When the event rates are not small,a maximum parsimony model may not be suitable any more.

With these assumptions, maximizing the four log likelihood functions can be related to the maximum parsimony optimization criterion. Of these four functions, one function can be represented as a lineage combination of event numbers and a function of corresponding event rates, which is the lineage likelihood model with birth-death event. The log likelihood function in the form of

$$\mathcal{L} = N_D f_1(r_D) + N_T f_2(r_T) + N_L f_3(r_L) + \text{Constant term.} \tag{5.101}$$

For event number likelihood with constant event rate, the log likelihood (Eqn. 5.97) can be separated into independent functions of $N_D, N_T$, and $N_L$, however, these are of the form:

$$\mathcal{L} = f_1'(r_D, N_D) + f_2'(r_T, N_T) + f_3'(r_L, N_L).$$

For the other two models (Eqn. 5.82, 5.90), the functions also contain the three linear terms as Eqn. 5.101, but also an extra $g()$ term that indicates interaction between gene events. The linear equation in the lineage likelihood model with birth-death event can be easily compared with MP optimization. If the cost vector $\vec{C}$ is selected as $< -f_1(r_D), -f_2(r_T), -f_3(r_L) >$, then maximizing the likelihood is the same as minimizing the cost function (Equation. 5.1). The extra $g()$ "interaction" term in event number likelihood with birth-death event and the lineage likelihood with constant event is an interesting area for future study to understand the amount of interaction and whether these likelihood functions can be also compared with the MP model. For the event number likelihood with constant event rate (Eqn. 5.97), if the term $\sum_s log(N_\epsilon(s)!)$ is neglibible, then the cost can be set to $-\log(r_\epsilon)$ and the MP and ML are equivalent. This is consistent with a proposal made by Felsenstein (2003)(page 99) for molecular phylogenetics.

# Chapter 6

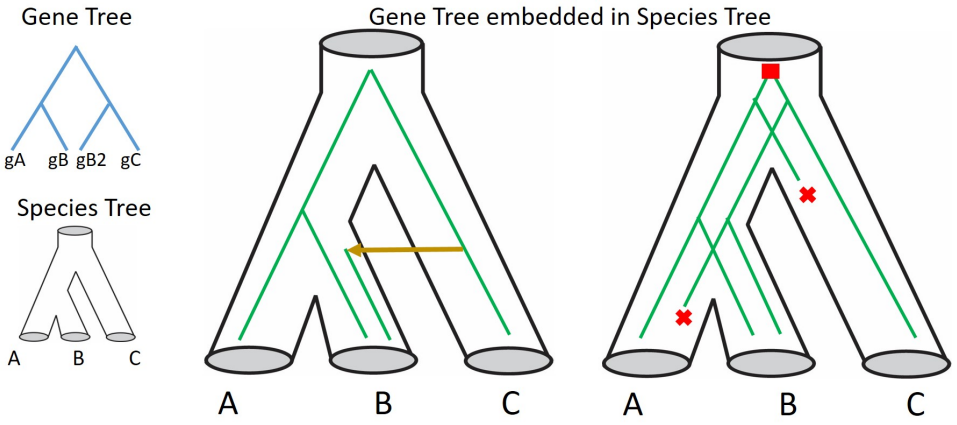
# Selecting Cost Using Cost Space Partition

Selecting event costs is a key aspect of carrying out an analysis using parsimony-based reconciliation. Recall that different cost vectors may have significant impact on the reconciliation solution. When analysing a single gene tree, the user can inspect the result and use biological intuition to assess whether the result is appropriate. However, for phylogenomic reconciliation, where thousands of gene trees are reconciled in an automated fashion, choosing the appropriate costs is essential for the final inference of gene events of all gene families.

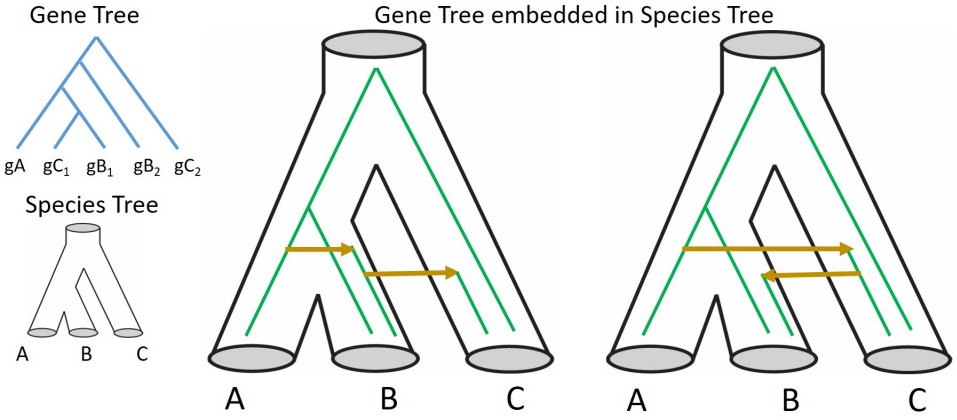
As we saw in Chapter 2, transfers increase the number of ways that the gene tree can be fit to the species tree. As a result, many different event histories can explain the same gene-species tree discordance. Reconciliation using different cost vectors may result in different most parsimonious solution. For example, the incongruence in Figure 6.1a can be explained by one transfer or by one duplication and two losses. If $(C_T, C_D, C_L) = (3, 1.5, 1.0)$, then one transfer is preferred. If $(C_T, C_D, C_L) = (4, 1.5, 1.0)$, then one duplication and two losses are preferred. Some histories may be more biologically realistic than others and the goal is to select a cost vector that gives a biologically meaningful solution.

The key challenge for users is selecting the cost vector that produces the best reconciliation. The user may have a qualitative sense of what types of event histories are plausible, but not be able to translate that intuition into specific numeric costs. Alternatively, the user may wish to compare event histories obtained with a representative set of cost vectors. This requires a set of cost vectors that broadly probes the space of parameters. The user could then select the results that seem most plausible, based on specialized knowledge, or combine

the results from several cost vectors in some way. For this approach to be efficient, the user requires that each cost vector used gives different results. For example, in Figure 6.1a the cost vectors $(3.0, 1.5, 1.0)$, $(3.0, 2.5, 1.0)$, and $(6.0, 3.0, 2.0)$ will all prefer the reconciliation with one transfer. Therefore, using all three cost vectors would be a waste of time.




(a)




(b)

Figure 6.1: Multiple solutions in the Maximum Parsimony framework. (a) Two reconciliation solutions that infers different number of events for the same gene tree. The left solution requires one transfer; the right solution requires one duplication and two losses. (b) Two reconciliation solutions that both require two transfer events, but the transfer events are different.

Another consideration is degeneracy. Some cost vectors will result in more optimal solutions than others. For example, with the cost vector $(C_T, C_D, C_L) = (3.5, 1.5, 1.0)$, both event histories in Figure 6.1a will be optimal. The cost vector can be chosen to reduce degeneracy. This observation suggests that the space of all cost vectors may be partitioned by some

function, so that all cost vectors in one region prefer one history and all cost vectors in another region prefer another history. For the example in Figure 6.1a, the cost space can be partitioned with a linear function (Figure 6.2a). However, it may not be possible to find a cost vector that corresponds to exactly one optimal solution. For example, Figure 6.1b shows two optimal solutions; both require exactly the same number of transfers. These two solutions are not distinguishable by selecting a specific cost vector. Therefore, choosing a good cost vector is crucial in multiple aspects: 1) A good cost vector infers biologically realistic results. 2) A good cost vector reduces the number of optimal solutions that must be inspected. 3) For phylogenomic analysis, by reducing the number of multiple optimal solutions, a good cost vector significantly speeds up the procedure for checking temporal consistency.

Here, I present a framework for selecting a set of cost vectors that depends on trade-offs between events of different types. Further, I present vignettes that illustrate specific scenarios that trigger such event tradeoffs. These vignettes represent basic elements that often arise in DTL-reconciliation with real trees. They also provide intuition about whether a given cost vector will result in a plausible explanation. I illustrate the impact of cost vector choice on accuracy, using reconciliation of simulated gene trees.

I first describe the idea of displaying the space of all cost vectors in a 2-dimensional plane. The plane can be partitioned into many regions, such that the cost vectors in each region gives the same result (Stolzer, 2012).

The goal of MP reconciliation is to find reconciliation solutions that have the lowest cost compared to other solutions. This suggests that the cost of one solution relative to another is more important than the actual value of the cost. For example, cost vectors $< C_T, C_D, C_L >$ and $< C_T/C_D, 1, C_L/C_D >$ will generate the exact same solutions. As long as the ratio stays the same, the values of each cost can be quite arbitrary. Without loss of generality, we will assume $C_D = 1$ for the rest of this chapter. Therefore, the space of all cost vectors can be represented as a two-dimensional space, where the values of $C_L$ and $C_T$ are represented on the horizontal and vertical axes, respectively. Every possible cost vector can be represented by a point in this 2D space.

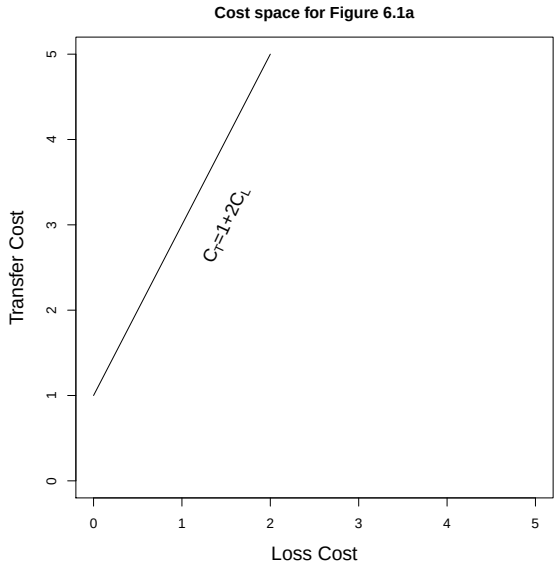
Given a gene tree, $T_G$, and a species tree, $T_S$, let $\overrightarrow{C^1} =< C_T^1, C_L^1 >$ and $\overrightarrow{C^2} =< C_T^2, C_L^2 >$ be two different cost vector and let $\Gamma^1$ and $\Gamma^2$ be the sets of optimal solutions for cost vectors $\overrightarrow{C^1}$ and $\overrightarrow{C^2}$, respectively. Then $\overrightarrow{C^1}$ and $\overrightarrow{C^2}$ are in the same partition if and only if for each

reconciliation $\gamma^1 \in \Gamma^1$, there exists a reconciliation $\gamma^2 \in \Gamma^2$, such that the number of events of each type is the same in both (i.e., $N_D^1 = N_D^2, N_T^1 = N_T^2, N_L^1 = N_L^2$) and vice versa.

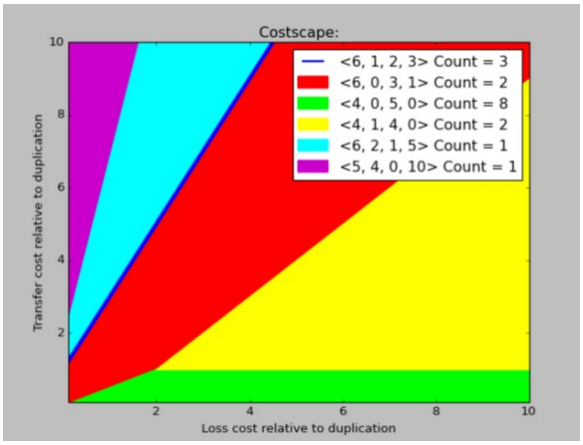
This statement of partition defines the problem of finding the cost space partitions for a given gene tree and species tree (Stolzer, 2012). For an analysis with several cost vectors, each cost vector should be associated with a different region in the partition. Cost vectors in one partition prefer one set of events and cost vectors in another partition will infer a different set of events. The line between two adjacent partitions indicates scenarios where both partitions overlap. Note this also indicates that cost vectors on the line will generate the union of the solutions from the adjacent partitions. It is even possible to observe several partitions converge at a point in the cost space. The cost vector corresponding to this point would result in the union of solutions from multiple partitions. By picking the cost inside a partition (instead of on a line or point), the number of generated optimal solutions is reduced and may provide a manageable set of solutions to be inspected. On the other hand, if the goal is to inspect as many solutions as possible, we could pick cost vectors on the line between partitions which will result in more optimal solutions

This problem has been tackled by Libeskind-Hadas et al. (2014), who developed an algorithm to reconcile the gene tree with species tree and split the cost space into several partitions. All cost vectors in the same partition give the same set of optimal solutions for a given gene tree and species tree (Libeskind-Hadas et al., 2014). The algorithm adapted the dynamic programming framework of regular DTL reconciliation. At each gene node, instead of calculating a cost table, this algorithm calculates an augmented cost table that stores the costs of combinations of events that have the potential to be part of the most parsimonious solution. This algorithm requires $O(|V_G|^5|V_S|\log(|V_G|))$ complexity to calculate a cost space partition for any gene tree and species tree. The output results in a figure similar to Figure 6.2b.

This algorithm is a step toward selecting cost vectors that reduce the degeneracy and is helpful for the exploratory analysis to inspect all solutions of a single gene tree. For phylogenomic reconciliation with one species tree and many gene trees, one approach would be to calculate the cost space partition for each gene tree and overlap these partitions. This would result in a combined partition of the cost space, representing the entire set of gene trees, which could result in a very large number of different regions. Because it is too computationally expensive to calculate the partition for every gene tree, instead, here, I propose a heuristic to summarize a cost space partition that works well for many trees, based on the species tree topology alone.




(a) Partition corresponding to Figure 6.1a




(b) Example output of Xscape

Figure 6.2: (a) The partition of cost space corresponding to Figure 6.1a. Each point in the space represents a cost vector. The space is separated by a linear function $C_T = C_D + 2C_L$ with $C_D$ is set to 1. Above this line, the solution with one duplication and two losses is favored; below the line, the solution with one transfer is favored.(b) Example of the output of Xscape(Libeskind-Hadas et al., 2014) for an example gene tree and species tree. Every cost vector in the same partition generates the same solution set. $< N_S, N_D, N_T, N_L >$ are the number of speciation, duplication, transfer, and loss events, respectively. Count is the number of optimal solutions obtained for any cost vector in the region.

By examining the 2D plane and gene tree topological structure, we discovered that many lines that split the 2D plane correspond to some specific, yet common, patterns of gene tree incongruence that are likely to occur in many gene trees, suggesting that some regions of the cost space may be suitable for most gene trees. Therefore, instead of generating the cost partition for each gene tree and combining them, our goal is to find a "plausible", common partition of the cost space before reconciliation that provides reasonable candidate regions to select multiple cost vectors.

Here, we categorize the common trade-offs between two event histories and present vignettes that illustrate specific scenarios that trigger such event tradeoffs. These vignettes represent basic elements that often arise in DTL-reconciliation with real trees. They also provide intuition about whether a given cost vector will result in a plausible explanation

| Scenario | This study |
|---|---|
| D versus T | here |
| D versus L | impossible |
| T versus L | here |
| D versus T & L | here |
| T versus D & L | complex |
| L versus T & D | complex |

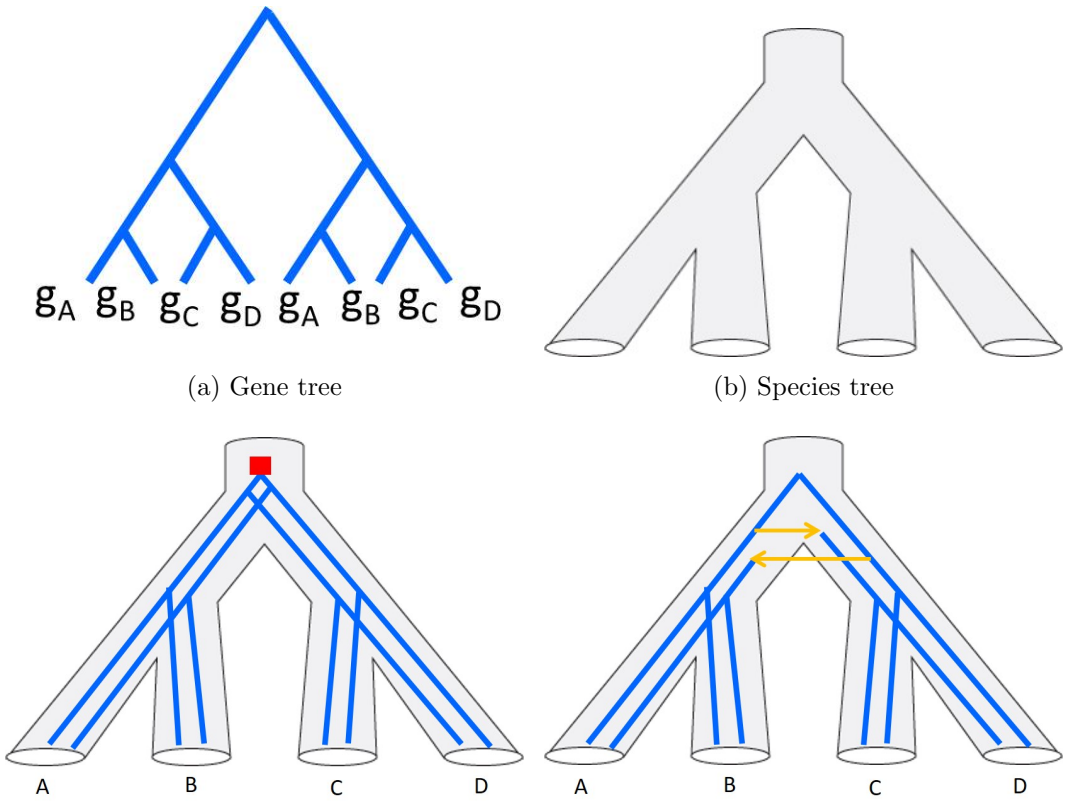
Table 6.1: Six scenarios of event trade-offs. The first three scenarios involve trading one event type for one other event type. The last three correspond to scenarios where one event type is replaced with some combinations of the two other event types. This study included three scenario as indicated. The remaining three scenarios are either impossible, or involve complex scenarios that may not arise frequently.

## 6.1. Trade-offs of different scenarios

Let $< N_D, N_T, N_L >$ be an event vector representing the number of duplications, transfers, and losses for one solution. The boundary between regions in the cost space represents a trade-off between two event vectors $< N_D^1, N_T^1, N_L^1 >$ and $< N_D^2, N_T^2, N_L^2 >$. From the perspective of a user, this can translate into a question of biological intuition, e.g. "Given my knowledge of these species, is a horizontal gene transfer more or less plausible than many losses?".

There are three possible cases where one event type can be traded for another and three cases where one event type can be traded for some combination of the other two event types (Table 6.1). Of these six scenarios, I focus on the three that are most plausible (Transfer versus Duplication, Transfer versus Loss, and Transfer versus Duplication and Loss). A scenario in which duplications can be exchanged for only losses cannot arise. Of the three cases involving exchanges of one event type for the other two event types, only one is likely to arise frequently (Transfer versus Duplication and Loss). The other two cases, if they arise at all, would require very skewed cost vectors in which the cost of one event was much larger than the costs of the two other events.

### 6.1.1. Duplication versus Transfer




(a) Gene tree

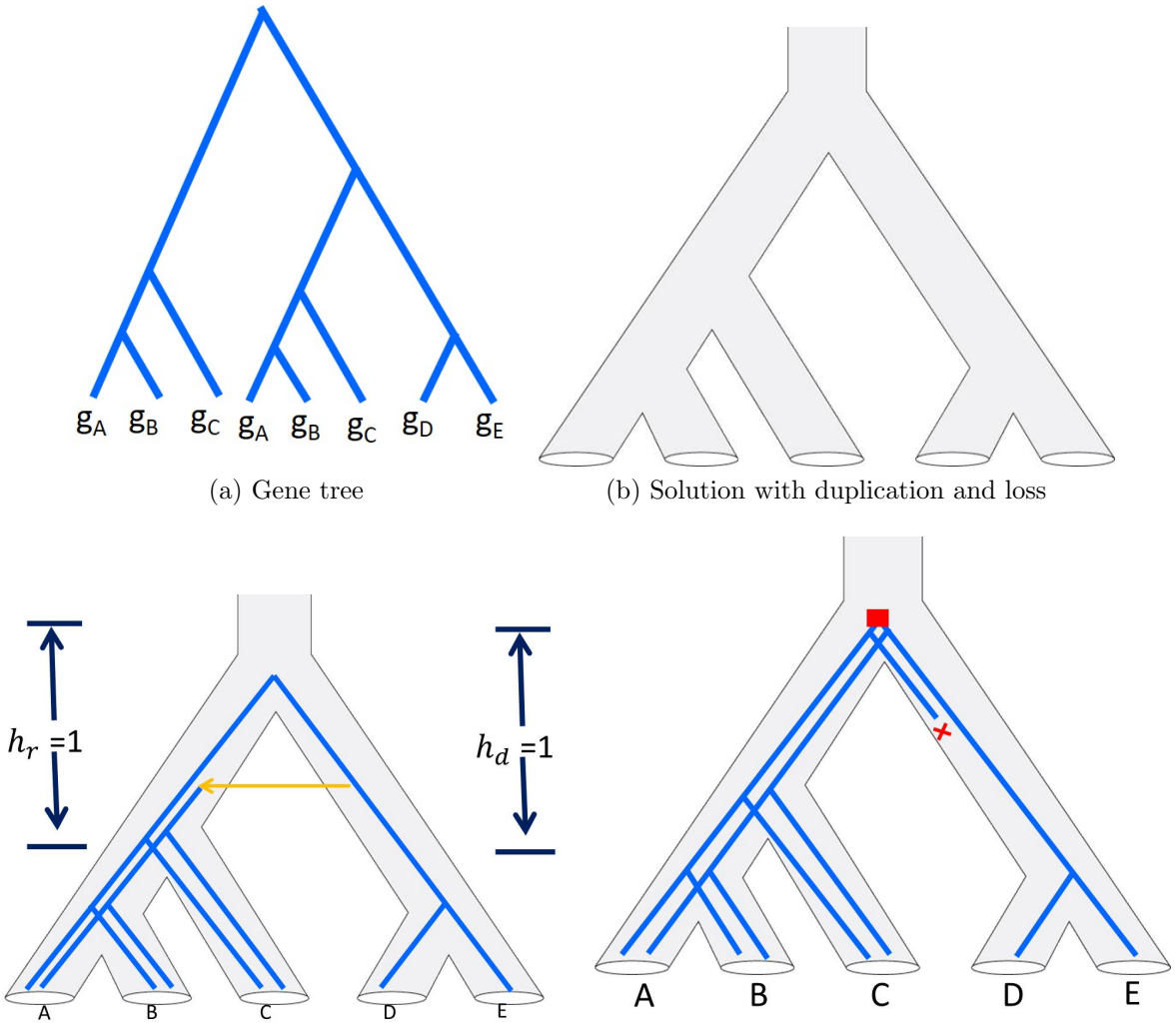
(b) Species tree

(c) Solution with one duplication

(d) Solution with wwo transfers

Figure 6.3: One duplication versus two transfers: when $2C_T < C_D$, any non-leaf duplication (c) can be traded for two transfers (d). For these costs, the reconciliation will not find any duplications on internal gene nodes

Gene tree incongruence that can be explained by a duplication event can always also be explained by two transfer events, as shoing in Figure 6.3. Therefore, when $2C_T < C_D$, any solution that requires $k$ duplications is not as parsimonious as a solution that requires $2k$ transfer events. The line $C_T = 0.5$ separates cost vectors that prefer $2k$ transfers from those that prefer $k$ duplications (recall $C_D = 1$). In some prokaryotes, cost vectors below this line may be a good choice of cost vector; otherwise, this cost space region may be unrealistic.



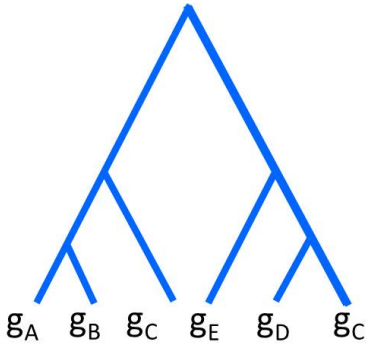

(a) Gene tree

(b) Solution with duplication and loss

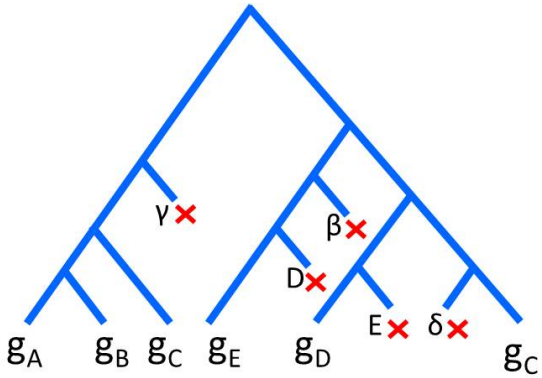
(c) Embedded solution with one transfer

(d) Embedded solution with two duplications and five losses

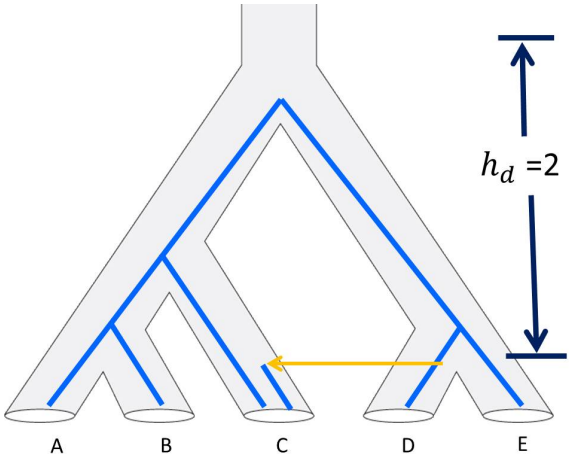
Figure 6.4: One transfer versus two duplications and five losses: When $h_d = 2$ and $h_r = 2$, a transfer is preferred if $C_T < 2C_D + 5C_L$. Two duplication and five losses is preferred if $C_T > 2C_D + 5C_L$.



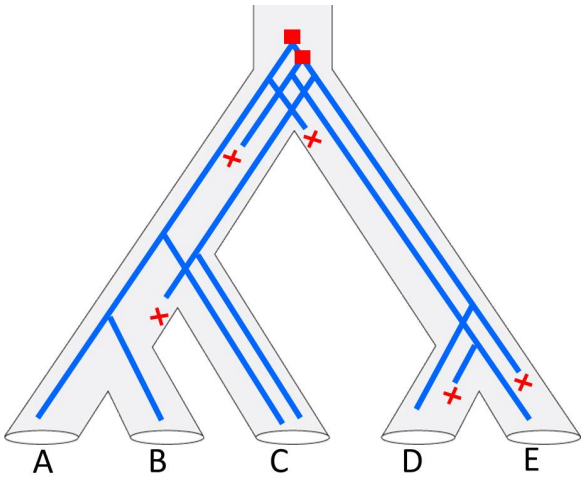

(a) Gene tree




(b) Solution with duplication and loss




(c) Embedded solution with one transfer




(d) Embedded solution with two duplications and five losses

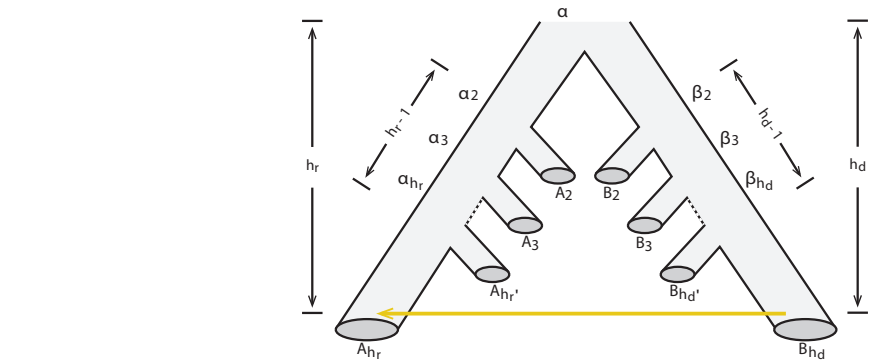
Figure 6.5: One transfer versus two duplications and five losses: When $h_d = 2$ and $h_r = 2$, a transfer is preferred if $C_T < 2C_D + 5C_L$. Two duplication and five losses is preferred if $C_T > 2C_D + 5C_L$.

### 6.1.2. Transfer versus Duplications and Losses

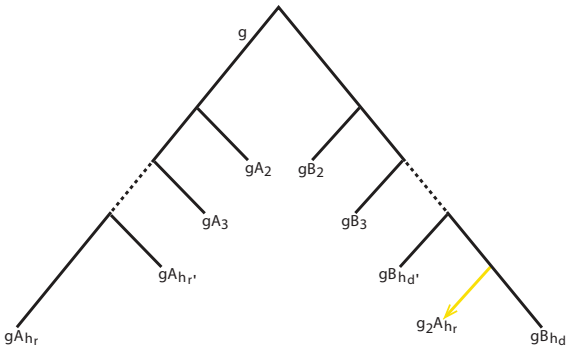
A transfer from species $s_d$ to $s_r$, and from gene $g_d$ to $g_r$, introduces a new copy of the gene into species $s_r$. If this copy is then inherited by the descendants of $s_r$ without further events, the transfer effectively introduces a copy of the subtree rooted at $s_r$ into the gene tree. The same topology could also occur if one or more duplications occurred in $s = \text{LCA}(s_d, s_r)$ followed by subsequent losses in these gene subtrees resulted in gene subtrees that are embedded in $T_s$, the species subtree rooted at $s$. In this scenario, the transfer can be traded for some number of duplications and losses as illustrated in Figure 6.5. The number of duplications and losses depends on the depth of $s_d$ and $s_r$ relative to $s$, denoted $h_d$ and $h_r$, respectively. In order to replace a gene transfer with duplication and losses, all the gene nodes that are ancestors of $g_d$ and descendants of $g = \text{LCA}(g_d, g_r)$ must reside in $s$, requiring that one or multiple duplications occurred in $s$. For example, in Figure 6.5, the donor gene is $g_D$, and the recipient gene is $g_C$. There are three gene nodes that must reside in $s$: $g_d$, $g$, and one node between $g$ and $g_d$. Therefore, two duplications occurred in $s$ to create three total lineages. The first lineage sustained 1 loss of the donor side and is retained only on the recipient side. The second lineage sustained 2 losses and is retained on the donor side. The third lineage that contains both $g_d$ and $g_r$ exists on both sides and sustained 2 losses, one on each side of $s$.

In Figure 6.6, the general rules are shown. The number of gene nodes inclusive of and between $g_d$ and $g$ is always $h_d + 1$, requiring that $h_d$ duplications created $h_d + 1$ lineages (showing in boxes with dashed lines) in species $s$. Of the $h_d + 1$ lineages, one lineage is retained only in species on the recipient side, $h_d - 1$ lineages are only retained in species on the donor side and one lineage is retained in some species on both sides, namely, $s_r$ and $s_d$ and their descendants. Specifically, the first lineage always sustains 1 loss and is retained only on the recipient side. For example, in Figure 6.6, the first lineage only exists in $T_{\alpha_2}$, therefore it sustained one loss at the root of the donor subtree. Each of the next $h_d - 1$ lineages is retained in a different subtree and only in that subtree on the donor side of the species tree. The second lineage only exists in $T_{B_2}$, and sustained 2 losses at the roots of subtree $T_{\alpha_2}$ and $T_{\beta_3}$. The third lineage only exists in $T_{B_3}$; 3 losses occurred in $T_{\alpha_2}$, $T_{B_2}$, and $T_{\beta_4}$. In general, the number of losses in these $h_d$ lineages increases by 1 for each additional duplication, forming an arithmetic sequence $(1, 2, 3, 4, ...h_d)$. The sum of those losses reduces to

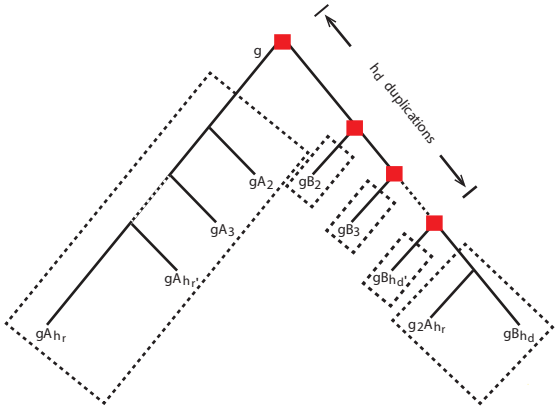
$$\frac{h_d^2 + h_d}{2}. \tag{6.1}$$




(a) Species tree




(b) Solution with a transfer




(c) Solution with duplications and losses

Figure 6.6: One transfer versus $h_d$ duplications and several losses. Given a transfer between $s_d$ and $s_r$ with depth $h_d$ and $h_r$. The transfer event can be replaced by combination of $h_d$ duplications and many losses. The yellow arrow represent a gene transfer, the red squares represent gene duplications.

The last lineage, which is present in descendants of both the donor and the recipient species implying $h_d - 1$ losses on donor side of the species tree $(B_2, B_3 ... B_{h_d-1})$ and $h_r - 1$ losses on recipient side of the species tree $(A_2, A_3 ... A_{h_r-1})$. Combining the losses that occurred in the first $h_d$ lineage (Eq. 6.1) and the losses in the last lineage, a total of

$$N_L = \frac{h_d^2 + 3h_d}{2} - 2 + h_r$$

losses will be required.

Therefore, for a transfer in a species tree with depths $h_d$ and $h_r$, the line

$$C_T = \left(h_r + \frac{h_d^2 + 3h_d}{2} - 2\right) \cdot C_L + h_d \cdot C_D, \tag{6.2}$$

separates cost vectors that prefer a transfer and cost vectors that prefer a combination of duplications and losses.

### 6.1.3. Transfer versus Loss

One approach to inferring horizontal gene transfer is via inspection of the phylogenetic distribution of the gene family in a group of species (Azad and Lawrence, 2012, reviewed in). If the species that possess copies of the gene family are sufficiently distantly related, then horizontal transfer is invoked as a more plausible explanation than massive loss. Consider a gene family which has family members in some, but not all, of the species represented in the species tree. This scenario could arise if the gene family were present in the common ancestor of those species and subsequently lost in some species lineages. Another possibility is that the gene family was first acquired in a more recent ancestral species (presumably by a transfer from a species distantly related to the species represented in the species tree) and subsequently transferred to an incomparable clade within the species tree.

This scenario introduces another trade-off (Fig 6.7). In general, the tradeoff between transfers and losses is defined by $nC_T$ versus $mC_L$, where $n$ is the depth of the first consecutive loss and $m$ is the total number of consecutive missing species. The line

$$nC_T = mC_L, \tag{6.3}$$

separates cost vectors that prefer $n$ transfers from cost vectors that prefer $m$ losses. The value of $m$ and $n$ are bounded by the height of the species tree.

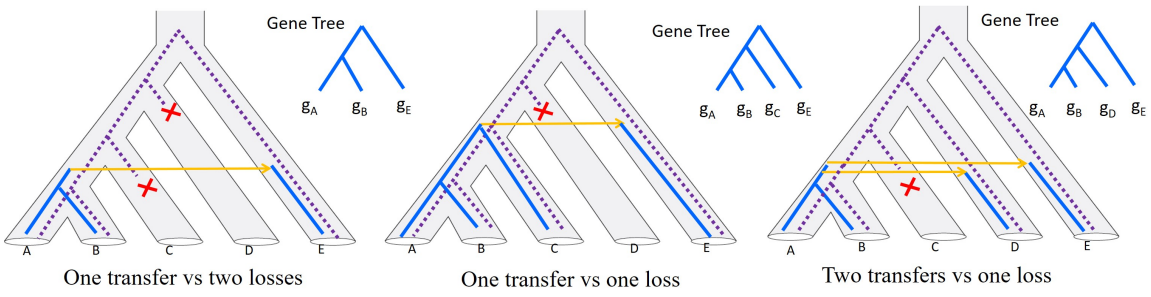



Figure 6.7: Three examples of a trade-off between transfer and loss events only. In each example, the purple embedded gene tree represent the reconciliation with loss events only, and the blue embedded gene tree represent the reconciliation with transfer events only. The $m$ and $n$ value of these three examples are :$m = 2, n = 1$, $m = 1, n = 1$, $m = 1, n = 2$.

### 6.1.4. Generic Cost Space partitions by Species Tree

The three trade-offs presented here describe several gene tree scenarios that are likely to arise in phylogenetic reconciliation. The gene trees in a phylogenomic analysis may very well contain many of these scenarios. A cost space partition based on these scenarios can be obtained from the species tree alone.

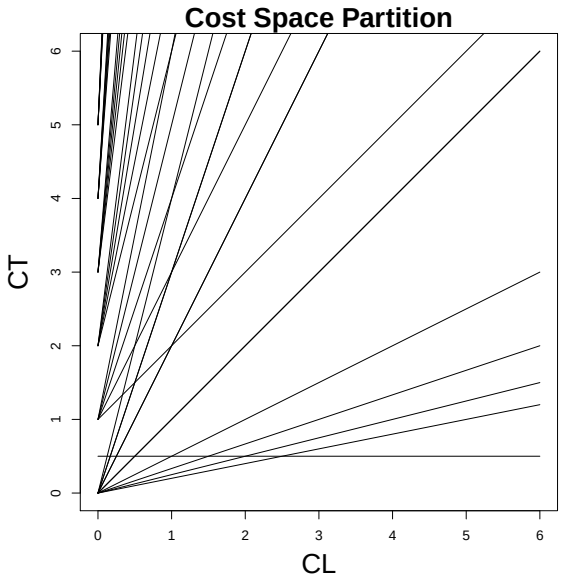



Figure 6.8: Partition of a 2D cost plane for a small species tree of height 6. The $C_T, C_L$ ranges are set to $[0, 6]$.

For each pair of $h_d$ and $h_r$ in the species tree, the "Transfer versus Duplication and Loss" trade off can be represented as a line, for which the cost vectors above the line prefer duplication and loss and the cost vectors below the line prefer transfer events. For a given species tree, let $\{(s_d^i, s_r^i)\}$ be the set of incomparable species pairs (i.e., the set of species pairs that could participate in a horizontal transfer) and let $h_d^i$ and $h_r^i$ be the depths of $s_d^i$ and $s_r^i$ relative to $\text{LCA}(s_d^i, s_r^i)$. Then each pair of $h_d^i$ and $h_r^i$ represents a trade-off (Eqn. 6.2). Note that many pairs of donor and recipient species may correspond to the same values of $h_r$ and $h_d$. In fact, if a pair of $(h_r, h_d)$ exist, and $h_r > 1, h_d > 1$, then two similar pairs, $(h_r - 1, h_d)$ and $(h_r, h_d - 1)$ also exist. This suggests that given the species tree height $h$, there exists $h \times h$ pairs that can be enumerated, each corresponding to a trade-off. Similarly, for each possible value of pair of $n$ and $m$ in the species tree, the "Transfer versus Loss" trade off can be represented as a line in the cost space graph. The cost vectors above the line $nC_T = mC_L$, represent the trade-off between loss events and transfer events only. The values of $m$ and $n$ are bounded by the height of the gene tree, which suggests there are also $O(h^2)$ pairs of $m, n$ values to be enumerated. Finally, a single line ($C_T = 0.5$) is added for the "Transfer versus Duplication" trade-off, which is independent from the height of species tree. Taken together, these three scenarios introduce many lines into the cost space and generally partition the cost space (Figure 6.8).

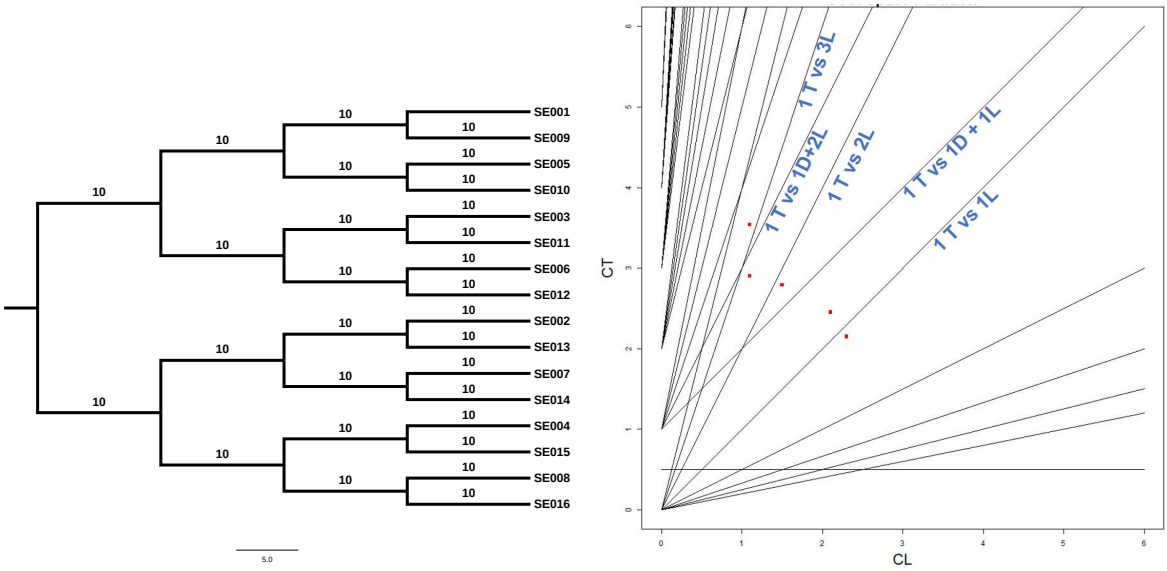
The major advantage of this approach is that the basic partition can be generated before the actual reconciliation of gene trees. These basic partitions are extremely convenient for phylogenomic analysis where many gene trees are reconciled with one species tree and can be used to select a set of different cost vectors for a phylogenomic analysis. Cost vectors in different partitions are likely to result in different reconciliations for many gene trees in the data set, although this is not guaranteed for a partition based on the species tree alone. Analysing these results will give insights to which events are robust and persist across different cost vectors, providing a more plausible interpretation than using a single cost vector. Further, these vignettes provide a visual representation of the meaning of different cost vectors, allowing users to pick costs that correspond to their biological intuition.

## 6.2. Simulation analysis of costs from five partitions

The species tree partition described in the previous section provides a guide for selecting cost vectors for DTL reconciliation, especially for phylogenomic analysis, where thousands of gene

trees are reconciled simultaneously. The basic partitions constructed for a species tree may provide good candidate cost vectors to be used in phylogenomic reconciliation. First, based on biological intuition, some of the candidate cost vectors may be more appropriate for a given set of gene tree and species tree. Second, analysis based on multiple cost vectors, each from a distinct partition, would reveal how parameter choice influences inference and makes it possible to identify trends that are robust. The analysis of the sub-optimal, or even "bad" cost vectors could lead us to understand the effectiveness of good versus bad cost vectors. To better understand these features of the cost space partition, I conducted a simulation study.

### 6.2.1. Experimental Setup




(a) Species tree used for simulation

(b) Five cost vectors

Figure 6.9: (a) Species tree used for simulation Each branch is normalized to 10 time units. (b) Five cost vectors (red dots) selected to represent different regions in the cost space partition. These cost vectors are listed in Table 6.3 in descending order.

The ALF simulation framework (Dalquen et al., 2011) was used to generate gene trees with various event rates on a balanced species tree with 16 leaves and uniform branch length (Figure 6.9). For every gene, the simulation starts at the root of species tree. This is a requirement of the software. Transfer, duplication, and loss events are generated based on

| $r_D$ | $r_T$ | $r_L$ |
|---|---|---|
| 0.0025 | 0.0025 | 0.001 |
| 0.0050 | 0.0050 | 0.002 |
| 0.0075 | 0.0075 | 0.003 |
| 0.0100 | 0.0100 | 0.004 |
| | | 0.005 |
| | | 0.006 |

Table 6.2: The four duplication rates, four transfer rates, and six loss rates used in the simulation. In total, there are $4 \times 4 \times 6 = 96$ combinations of event rates. For each combination, 200 gene trees were generated.

a birth-death model with multiple combinations of event rates $\vec{r} =< r_T, r_D, r_L >$. Four duplication rates, four transfer rates, and six loss rates where used (Table 6.2). The time between the root and leaf nodes in the species tree is 100 time units. Assuming there are no more than 2 birth events per species tree branch, we select the rate of duplication and transfer events from 0.0025 to 0.01 so that the average number birth events for a gene family does not exceed 100, except for a few gene families. The loss event rates are selected to range from 0.001 to 0.006. The range of loss rate is less than the range of birth rate because a high loss rate would result in a large subset of simulated gene families that go extinct before reaching the end of the species tree. For each of the 96 combination of rates, 200 gene trees were generated, resulting in a total of 19200 gene trees. In the cost space, each line bipartitions the space based on the trade-off equation. This suggest that the reconciliation results of two different cost vectors ($\overrightarrow{C} =< C_T, C_D, C_L >$) from two partitions that are separated by a line should reflect this trade-off. The $19,200$ simulated gene trees were reconciled under DTL event model with five sets of costs (Table 6.3) selected from the cost partitions in Figure 6.9b, resulting in $96,000$ reconciliations. These costs were selected in different partitions and each exhibits a preference of a trade-off. For example, the cost vector $< 2.45, 1, 2.1 >$ is in the region bounded above by $C_T = C_D + C_L$, and bounded below by $C_T = C_L$, corresponding to the tradeoffs in Figure 6.4 and Figure 6.7(middle), respectively. With this cost vector, a transfer would be preferred to a duplication and a loss, but would not replace a loss alone. As the trade-offs that are involved in fewer events are simpler and more common to observe than large trade-offs with multiple events, we focus on the cost vectors in these partitions that involved fewer events.

| $C_T$ | $C_D$ | $C_L$ | $\Delta^{\vec{C}}(T)$ | $\Delta^{\vec{C}}(D)$ | $\Delta^{\vec{C}}(L)$ |
|---|---|---|---|---|---|
| 3.54 | 1 | 1.1 | 0.30 | 0.23 | 1.24 |
| 2.9 | 1 | 1.1 | 0.18 | 0.13 | 0.60 |
| 2.79 | 1 | 1.5 | 0.15 | 0.11 | 0.49 |
| 2.45 | 1 | 2.1 | 0.09 | 0.07 | 0.47 |
| 2.15 | 1 | 2.3 | 0.12 | 0.07 | 0.62 |
| | | | 0.22 | 0.1 | 0.68 |

Table 6.3: Table of five cost vectors covering five different regions in the cost space and the mean event error associated with each cost vector.

### 6.2.2. Result

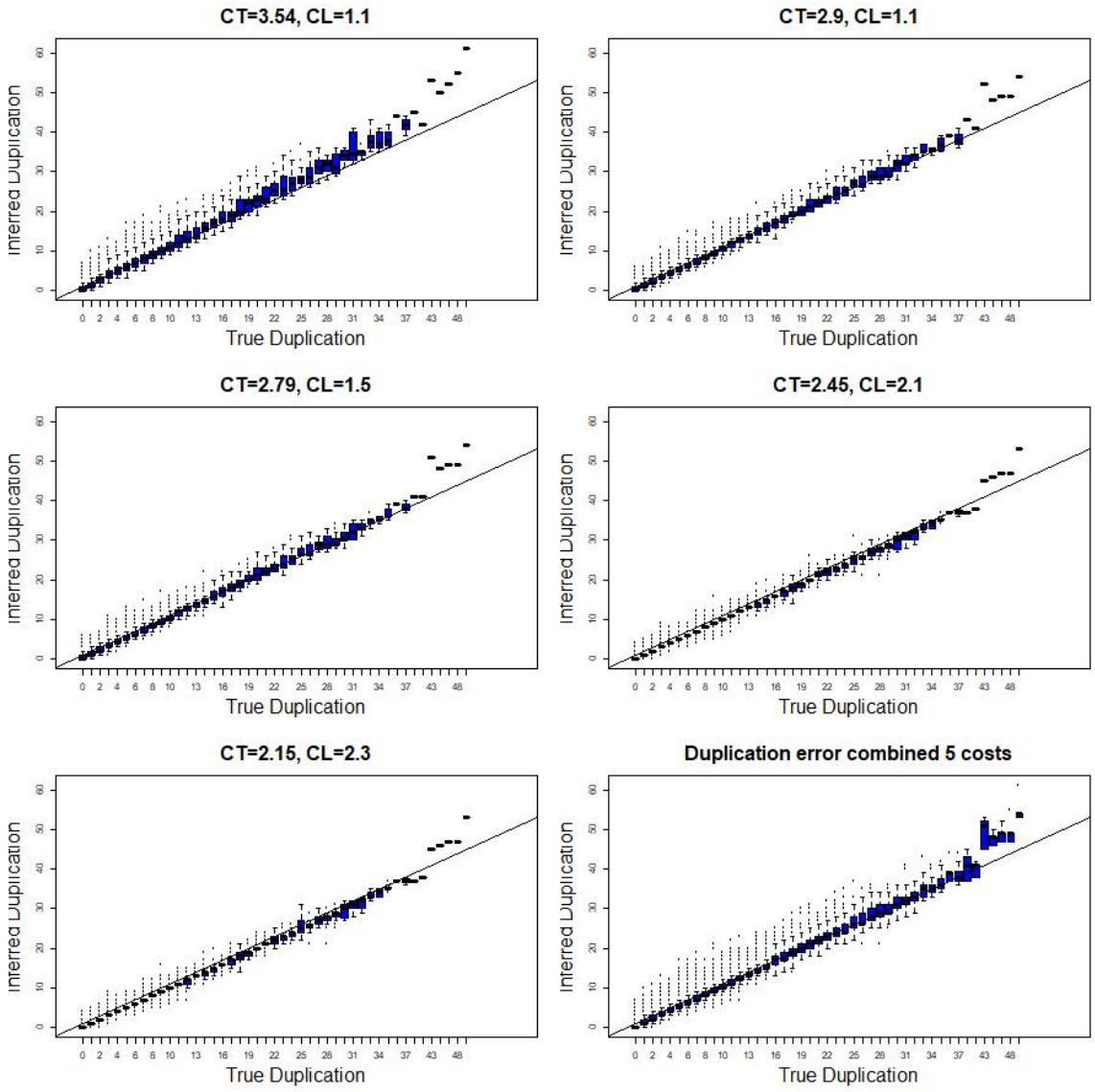





Figure 6.10: Inferred duplications versus true duplications for each cost vector.

The 19, 200 gene trees represent a mixture of different event rates, which is what would be expected from a collection of gene families sampled from the same species. Therefore, understanding how well each cost vector performs for gene families with gene families with mixed event rates is crucial for phylogenomic reconciliation analysis. We consider the accuracy of event inference in the entire data set of 19, 200 trees, combined, for each of the five cost vectors. For each of the three event types, I compared the number of inferred events and the number of true (i.e., simulated) events in each reconciled tree.

To have a quantitative assessment, the performance of each cost vector was evaluated using a series of error equations. Given a gene tree $T_G$, I define the error of event type $\epsilon$ for a gene family $g$ to be:

$$\Delta_g(\epsilon) = \frac{|\widehat{N_\epsilon} - N_\epsilon|}{N_\epsilon}, \tag{6.4}$$

where $\widehat{N_\epsilon}$ and $N_\epsilon$ are the inferred and true numbers of event of type $\epsilon$ respectively. I further define the mean error over 200 gene trees for a given combination of rates ($\vec{r} =< r_T, r_D, r_L >$) and a cost vector ($\overrightarrow{C} =< C_T, C_D, C_L >$) to be:

$$\Delta_{\vec{r}}^{\overrightarrow{C}}(\epsilon) = \frac{\sum_{i=1}^{200} \Delta_{g_i}(\epsilon)}{200}.$$

Similarly, a global error that describes the performance of the cost vector for all gene trees, over all rates, is calculated as:

$$\Delta^{\overrightarrow{C}}(\epsilon) = \frac{\sum_{r_T} \sum_{r_D} \sum_{r_L} \Delta_{\vec{r}}^{\overrightarrow{C}}(\epsilon)}{96}.$$

#### 6.2.2.1. Inferred events Compared to True events

Figures 6.10, 6.11, and 6.12 show the number of inferred versus true (i.e. simulated) duplications, transfers and losses respectively, for each of the five cost vectors (19200 reconciliations each) and for overall cost vectors (total 96000 reconciliations). Each box and whisker plot represents the distribution of inferred event counts ($N_\epsilon$), for a fixed true event count.

Of the three event types, the inference of duplications is most accurate. Transfer inference is also relatively robust. Except for the highest event cost ($C_T = 3.54$), the box and whisker plots are, for the most part, centered on a diagonal line with a slope of 1, which represents the situation where the number of inferred duplications is equal to the number of true

duplications (Figure 6.10). With very high transfer costs, the number of duplications tends to be overestimated, especially in trees in which a large number of duplications occurred. For any given cost vector, the number of inferred duplications is least accurate when the true duplication number is greater than 40. For a species tree with 16 leaves, $N_D > 40$ indicates that on average, 1.3 duplications occurred on every species branch, which is not very likely in real gene trees.

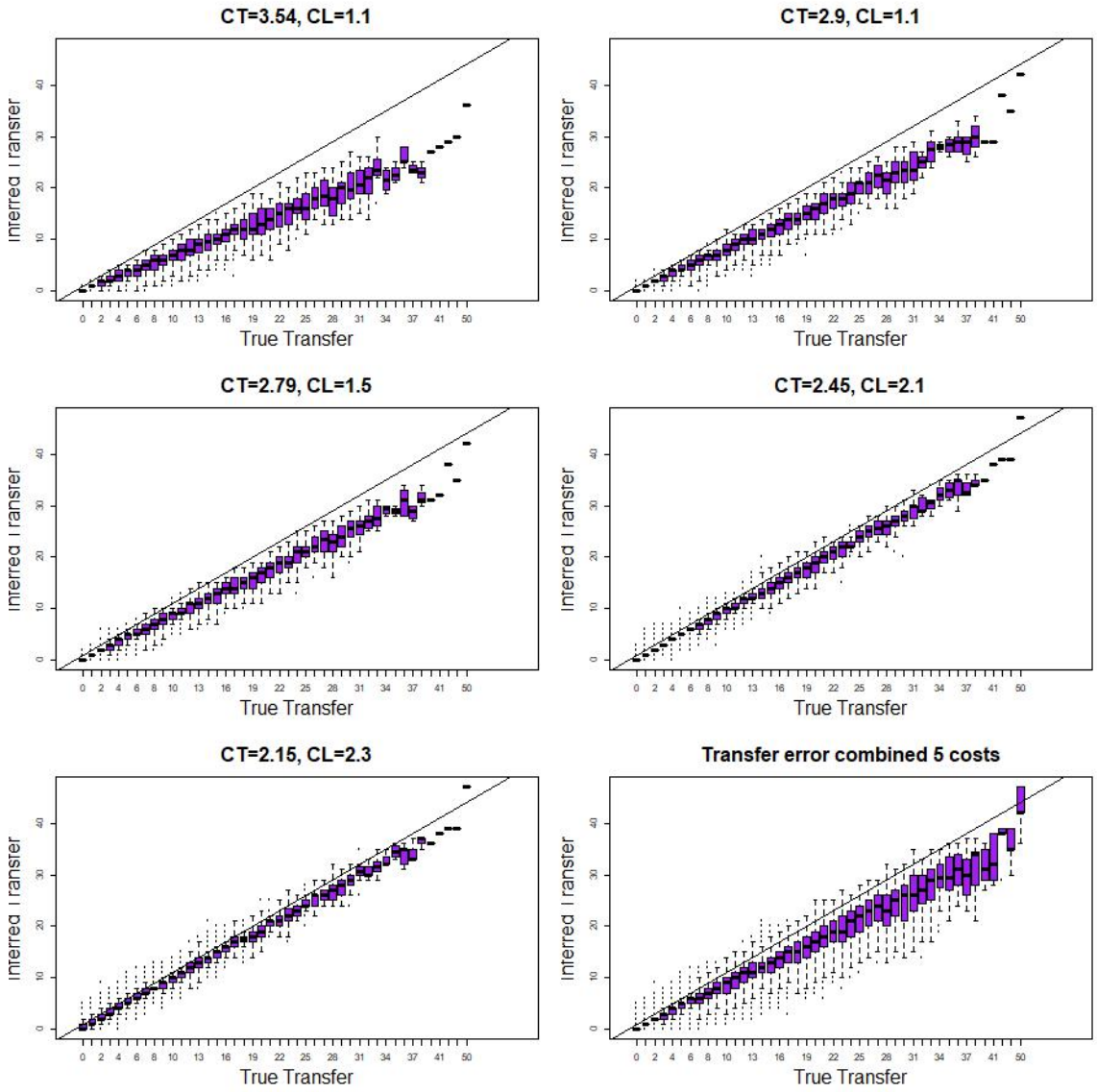



Figure 6.11: Inferred transfers versus true transfers for each cost vector.

This observation is consistent with the property of gene duplication. If no subsequent events

occur, a duplication on an internal node, $g$, will create two identical copies of the species tree subtree rooted at $\mathcal{M}[g]$. It is hard to imagine a combination of transfer and loss events could generate two such identical subtrees. This very specific pattern is unlikely to lead to an incorrect inference involving transfers and losses, instead of a duplication.

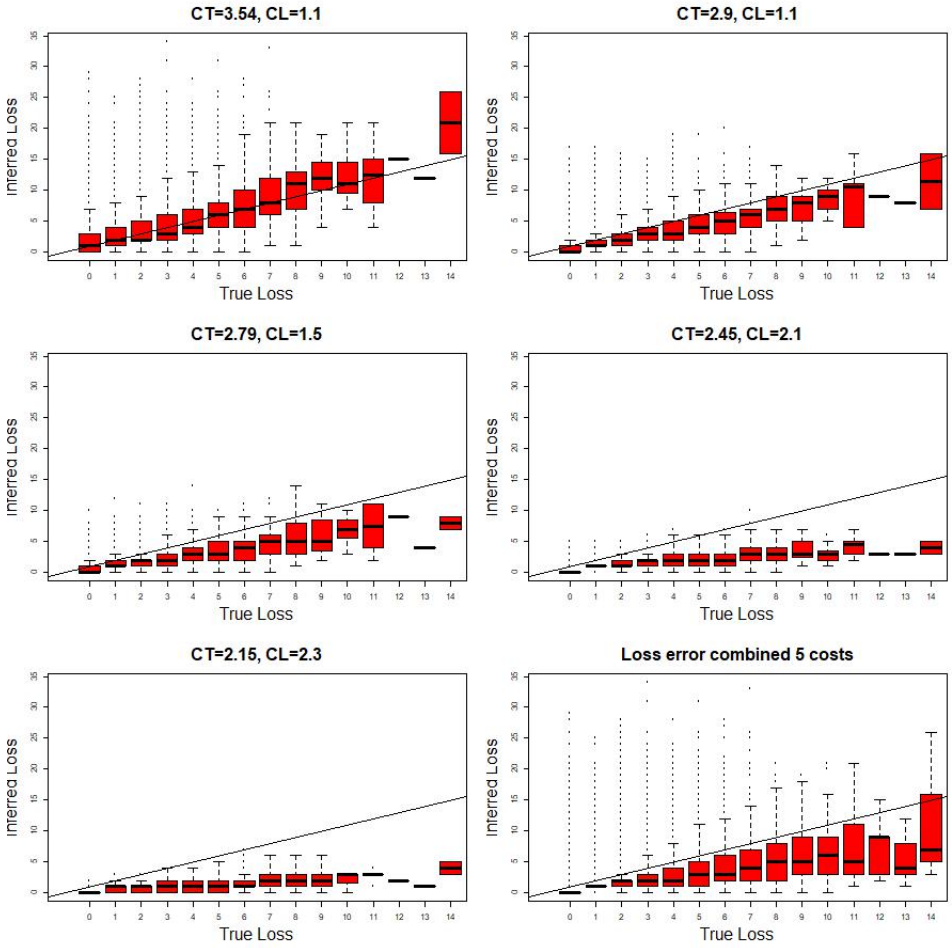



Figure 6.12: Reconciled losses versus true Losses number for each cost vector, and for all 96,000 reconciliation results. Box-whisker plot for each True Loss number.

Consistent with this, of the three event types, the global error associated with duplications is lowest. Figure 6.11 shows that the number of transfers is frequently underestimated and this effect is strongest with high transfer costs. This is consistent with the fact that, as we move from higher to lower partitions, the tradeoffs increasingly favour transfers over other events.

Losses are also underestimated, but the trend is in the opposite direction. Losses are least

| Factor | Sum Sq | p-value |
|---|---|---|
| rT | 0.007 | 1.87e-01 |
| rD | 0.168 | 8.02e-10 |
| rL | 0.614 | 2.00e-16 |
| cT | 5.025 | 2.00e-16 |
| cL | 0.194 | 4.32e-11 |

Table 6.4: ANOVA: factors that influences transfer error.

underestimated, on average, with the cost vector located in the highest region of the partition (Figure 6.12). In this partition, three losses or one duplication and two losses are preferred to one transfer. However, the error is highest with this cost vector, indicating a high variance in the number of losses inferred.

#### 6.2.2.2. Error Analysis

For each of the 96 combinations of $r_D, r_T, r_L$, 200 simulated gene trees were reconciled with 5 different cost vectors, resulting in 480 (96 rate vectors $\times$ 5 cost vectors) sets of reconciled trees, each with 200 gene trees. For each set, we calculated a error for three event types and generated histograms of all 480 errors. Figure 6.13 shows the error for each event type over all cost vectors and all rate combinations. When comparing the extremes of the three distributions, the losses have the highest errors, suggesting that losses are hardest to infer correctly. For a generally good reconciliation, one goal may be to find a cost set that minimizes loss error as the first priority.

To analyse what is the main contributor to the error rate of each event type, I carried out a five-way ANOVA analysis in which the event rates and costs of transfer and loss are the independent variables and error rates are the dependent variables (Table 6.4 and 6.5). Surprisingly, transfer error is not affected by transfer rate ($p - value = 0.187$), but is very significantly affected by the rate of duplication and loss and the cost of transfer. Loss error is not significantly affected by duplication rate ($p - value = 0.232$), but is very significantly affected by $rT, rL$, transfer cost and loss cost. We observe a negative correlation between $C_T$ and loss error, but a positive correlation between $C_T$ and transfer error. This indicates that it is hard to find a cost set that minimizes both loss error and transfer error; therefore, the best compromise between the errors needs to be considered for any particular analysis.

| Factor | Sum Sq | p-value |
|---|---|---|
| rT | 3.34 | 1.28e-08 |
| rD | 0.14 | 2.32e-01 |
| rL | 5.08 | 3.65e-12 |
| cT | 33.40 | 2.00e-16 |
| cL | 9.69 | 2.00e-16 |

Table 6.5: ANOVA: factors that influences loss error.

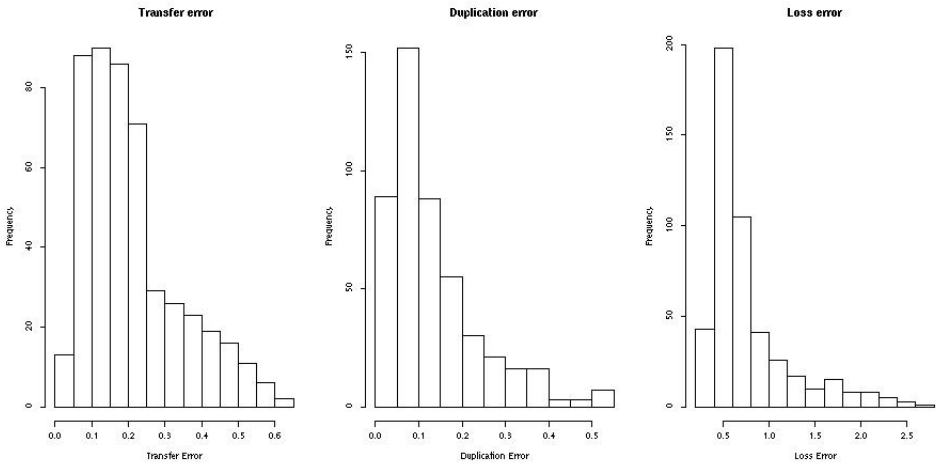



Figure 6.13: From four duplication rates, four transfer rates, six loss rates, five cost sets, 480 groups of reconciliation results were obtained. The histogram shows the T,D,L error rate for each group.

#### 6.2.2.3. Discussion

The error from loss events is the major contributor to the total reconciliation error. This suggests that the inference of loss events requires further investigation. While the other two event types are more accurately inferred, generally underestimating the number of loss event (Figure 6.12) suggests that the event rates used in the simulation may be not suitable for the MP framework. Because the MP cost is almost always less than the true cost calculated from the event history, the difference between these two costs is mostly the difference of loss events. This suggests the inferred duplication and transfer events may be lower (closer to a leaf than the root) than the true position and fewer loss events are inferred.

The analysis in this chapter is based on three simple scenarios that invoke a tradeoff between one or two event types. How often are more complicated scenarios at play? For any two

adjacent regions that are separated by a trade-off scenario in the cost space partition, the reconciliation result should reflect the simple trade-off scenario. For example, for the trade-off equation $C_T = C_L$, the total number of transfers and losses inferred for both cost vectors should be a constant, because the number of scenarios in the set of gene trees is fixed. Therefore, if the reconciliation using a cost vector above the trade-off equation results in $x$ transfers and $y$ losses, and using a cost vector below the trade-off equation results in $x - k$ transfers, then it must also infer $y + k$ losses. This would indicate that $k$ occurrences of this trade-off scenario exist in the group of gene trees. However, more complex scenarios in the gene trees may exist and therefore render the sum to be different. Therefore, comparing the reconciliation results of cost vectors from different partitions provides insights for how many of these complex scenarios exist in the set of gene trees such that the basic partitions are not enough to categorize all the trade-offs. A comparison of the results from different cost vectors that are only separated by one trade-off scenario would reveal the common trade-off scenarios between adjacent regions. The comparison of inferred gene events is a direct indicator of how many of the scenarios exist in the batch of gene trees, and may also indicate how many "complex" scenarios are not considered. This gave us insights to the advantage and compromise of selecting cost vectors from the basic partitions.

# Chapter 7

# Phylogenomic Reconciliation Using Notung

The algorithms that I have described in previous chapters (Chapter 3, 4) have been implemented in NOTUNG. NOTUNG is a Java based reconciliation software, that has been maintained and distributed by the Durand Lab (http://www.cs.cmu.edu/ durand/Notung/). The first prototype was developed in 2000 (Chen et al., 2000), and it has been made publicly available since 2005 (Durand et al., 2005). Notung contains a GUI mode, in which researchers can reconcile a single gene tree and observe reconciliation results and explore different cost sets and solutions. It also contains a command line mode, in which large number of gene trees can be processed efficiently.

My contribution to the NOTUNG software package includes: a) algorithmic advances described in Section 3.1, Section 3.2 and Chapter 4 in NOTUNG 2.9, b) implementation of algorithm described in Section 3.3 in NOTUNG-DM, c) more efficient diagnostic and memoization that speed up the runtime of software, d) on going participation in the software maintenance and responding to user inqueries and bug reports, e) functionalities that facilitate non technical NOTUNG users for phylogenetic analysis, such as GUI modification to iterate multiple optimal solutions, f) adding a phylogenomic mode for NOTUNG that is crucial for batch model phylogenomic reconciliation.

The NOTUNG software package was used in numerous phylogenomic analyses. The studies that were involved in application of NOTUNG software in 2016 include:

- Correlating enzyme evolution with geological events to date the emergence of the first

multicellular organisms (Gold et al., 2016).

- The evolution of the chemosensory repertoire: chemical communication in clonal raider ants (McKenzie et al., 2016), nocturnal vision in mammals (Kim et al., 2016a); opsins in the Bilaterian ancestor (Ramirez et al., 2016); and courtship pheromones in amphibians (Van Bocxlaer et al., 2016).

- Co-evolution of decay enzymes and saprophytic lifestyles in fungi (Druzhinina et al., 2016; Nagy et al., 2017; Varshney et al., 2016; Wisecaver et al., 2016).

- The emergence of parasitism: Adoption of an invasive lifestyle in D. suzukii (Hickner et al., 2016); expansion of digestive enzymes enabling predatory efficiency in spiders (Fuzita et al., 2016); host associations mediated by small secreted proteins in fungi (Kim et al., 2016b); and endoparasitism in nematode-trapping fungi (Zhang et al., 2016).

- Evolution of developmental programs in plants: evolution of leaves (Vasco et al., 2016); regulation of flowering (Zheng et al., J. Systematics Evol.); photoperiod sensitivity in maize (Li et al., Genomics); evolution of a plant defense system (Cao et al., Plant Growth Regulation); and expansion of loci encoding long intergenic noncoding RNAs (Nelson et al., G3).

NOTUNG is also being used in bioinformatics resources developed by other teams, including a database to support comparative genomic analysis in Nicotiana attenuata (Brockmöller et al., 2017) and GENVO (http://genvo.azurewebsites.net), a 3D visualization system for phylogenetic trees (Leandersson, Master's thesis, Stockholm U).

Within my own department of Biological Sciences at Carnegie Mellon University, I collaborated with Professor Luisa Hiller on a project to discover conjunctivitis virulence factors in *Streptococcus pneumoniae*. Using comparative genomics, Dr. Hiller and members of her laboratory identified genes that encode candidate virulence factors for colonization of the human eye by *S. pneumoniae*, including a predicted surface exposed agglutinin receptor protein (AR1). A deletion mutant of AR1 displayed reduced adherence to epithelial cells. I used NOTUNG to investigate the evolutionary origins of this virulence factor. The evidence from my reconciliation analysis supports the hypothesis that AR1 was acquired by lateral gene transfer from the pig pathogen, *Streptococcus suis*. This is an example of a medical application of phylogenetic reconciliation. A manuscript describing this work, submitted to

mSphere, is currently under revision in response to initial reviews.

In this chapter, I describe the phylogenomic mode reconciliation that I implemented in the Notung software package. This mode enables reconciliation with phylogenomic data and produces summary files of the reconciliation. Then I apply this model to a set of cyanobacteria gene family trees and demonstrate that reconciliation analysis in this model is straightforward and easily parsable to discover interesting biological insights.

## 7.1. Notung Phylogenomics: Reconciliation of Genomic Data

The advancement of sequencing technology has provided whole genome sequence data for a rapidly increasing number of species during the past decade. As a result, large collections of gene trees can be constructed for reconciliation analysis. Reconciliation of a single gene family tree provides a detailed history of a particular family. Modern genomic analyses often call for reconciliation of many gene families to understand other phenomena such as whole genome duplication, gene family co-evolution, genome streamlining, and patterns of horizontal gene transfers.

In phylogenomic reconciliation studies, the focus is on aggregating information associated with species nodes and branches, such as number and size of gene families associated with an ancestral species, and gene family expansion or contraction along each species tree branch. This information can be collected effectively as a table, in which rows represent each gene family and columns represent each species node or branch. The current approach for this type of analysis is to reconcile each gene tree individually and then parse the result of each reconciliation to generate the above tables. This requires a significant amount of computational expertise on the part of the user to convert the results into a form that can be analysed in different statistical software.

For this purpose, I developed a phylogenomics mode in NOTUNG. In this mode, NOTUNG can reconcile thousands of gene trees and generate summary files of reconciliation statistics. For each gene family tree, NOTUNG collects the number of duplications, losses, transfers received, transfers donated, gene family members gains and losses for each species node/branch.

The tables generated from the software can be output in formats easily read by analysis

software such as Excel, R etc. This implementation should significantly facilitate genomic level analysis and encourages non-technical users and users from differing fields, to easily analyse batch of trees with a single command.

### 7.1.1. Output of Phylogenomic Reconciliation

NOTUNG's phylogenomc reconciliation mode, activated by *–phylogenomic* flag in command-line, not only generate all reconciled gene trees, but also summarize the event information in eight tables. In these tables, each row represents a gene family and each column represents a species node (leaf and internal). These tables are summarized below:

1. Gene family origins:
   A file ending with `.origin.txt` contains the information of the origin of each gene family. The origin is defined as the species node where the gene family first appeared. In technical terms, the origin of a gene tree is the species node in which the gene tree root resides.

2. Ancestral gene content:
   A file ending with `.geneCount.txt` contains the number of gene copies of each gene family in each species node. A cell $i$, $j$ in the table represents the number of copies of gene family $i$ in species node $j$.

3. Gain and loss of gene families:
   A file ending with `.famGainLoss.txt` contains the changes of gene family in each species node. This file aims to find the families that went extinct or originated in a given species branch. Given a cell $i$, $j$ in this table, the number is 1 if the family originated in species $j$, 0 if the family exists in both species $j$ and its parent $p(j)$, or $-1$ if the family went extinct right before species $j$.

4. Net change in gene family size:
   A file ending with `.geneGainLoss.txt` reports the net change of gene copy of each gene family and for each species node.

5. Duplications:
   A file ending with `.duplication.txt` reports the number of duplications of each gene family in each species node.

6. Losses:
   A file ending with `.loss.txt` reports the number of losses of each gene family in each species node.

7. Transfers:
   A transfer event occurs between two species. Therefore we use two files: `*.transferFrom.txt` and `*.transferTo.txt` to summarize both outgoing and incoming transfers of each gene family and species node. A cell $i$, $j$ in table `*.transferFrom.txt` represents the number of outgoing transfers occurred in gene family $i$ from species node $j$. Similarly, a cell $i, j$ in table `*.transferTo.txt` represent, in gene family $i$, the number of incoming transfers, whose recipient is species node $j$.

8. Transfer highways:
   The file `*.highway.txt` does not follow the same format as the previous files, instead, for species tree with $n$ total species nodes, this file is a $(n+1)\times(n+1)$ matrix in comma separated format. This highway file contains a header row, and a side column denoting each species and each cell is the total number of transfers over all gene families, between pairs of species.

Ancestral species information can be calculated directly from these files. For example, the ancestral gene number can be calculated by simply summing over all rows in the `*.geneCount.txt` file. The net change of gene copy and gene families can be calculated similarly. The total number of gain of gene copies in a species node is the sum of duplication, incoming transfer, and origin that occurred in a species node, and the number of loss is simply the sum of loss event inferred in a species node. From `*.famGainLoss.txt` the researcher can easily find out the families that appeared or went extinct in a species branch. This would help them focus on a set of genes to understand how a group of species gained or lost abilities.

## 7.2. Case study: analysis ancestral gene content in Cyanothece

Here, I exemplify the power of the phylogenomics option in NOTUNG in an analysis of gene families in the distributed genome of nine Cyanothece species (Phylum Cyanobacteria). The

distributed genomes are defined as the collection of gene families, that exists in at least one of the nine Cyanothece species but not all of the Cyanothece species. Cyanothece are important model organisms with relevance to toxicology and the biofuel industry. We used the phylogenomics output from NOTUNG to analyze the patterns of lateral transfers, and to quantify gains and losses. Then we identified specific losses from species *mae* using enrichment analysis. Our analysis shows how NOTUNG can be used to identify and explore the trends in evolution and how they are related to ecology.

### 7.2.1. Pre-Processing: Distributed Gene Families and Core Gene Families

For this analysis, I used the Barker data set described in Section 4.4. The data set contains 13,854 gene trees that can be downloaded from http://dx.doi.org/10.7488/ds/1485. This set of gene trees is broad and might be erroneous because some of these clusters are small clusters with only one or two members among the Cyanotheces. We excluded the gene trees with only 3 leaves or fewer and kept the gene families that exist in one of the 9 cyanotheces but not all cyanotheces. This results in 5404 gene families as the distributed genome. Using NOTUNG 2.9, the 5404 full gene trees were rooted under DTL event parsimony and rearranged with DL event parsimony. I reconciled the resolved and rooted gene trees with phylogenomic mode and used the output to infer the ancestral gene family content, as well as the number of families gained and lost on each branch. For comparison, we inferred the same quantities with Wagner Parsimony using Count (Csurös, 2010). The ancestral family counts inferred using NOTUNG 2.9 and Wagner parsimony are very different (Figure 7.1). The Wagner parsimony only considers the number of gene copies in the leaves of the species tree, which suggests that the algorithm prefers to infer gene families to exist closer to the leaves. It results in a much smaller number of gene family exists at the root of the cyanothece species compare to the phylogenomic reconciliation result. Because the gene families are inferred to have originated more recently in the species tree, the Wagner parsimony also underestimates the degree of gene family gain and loss and suggests a history of gradual genome expansion. By taking advantage of phylogenetic information, NOTUNG 2.9 can infer more nuanced evolutionary histories: NOTUNG 2.9 infers an ancestral expansion of 2500 families, followed by genome reduction in *cwa* and *mae*. Family gains and losses inferred by NOTUNG 2.9 suggest ongoing gene family turn-over, consistent with high genome plasticity.

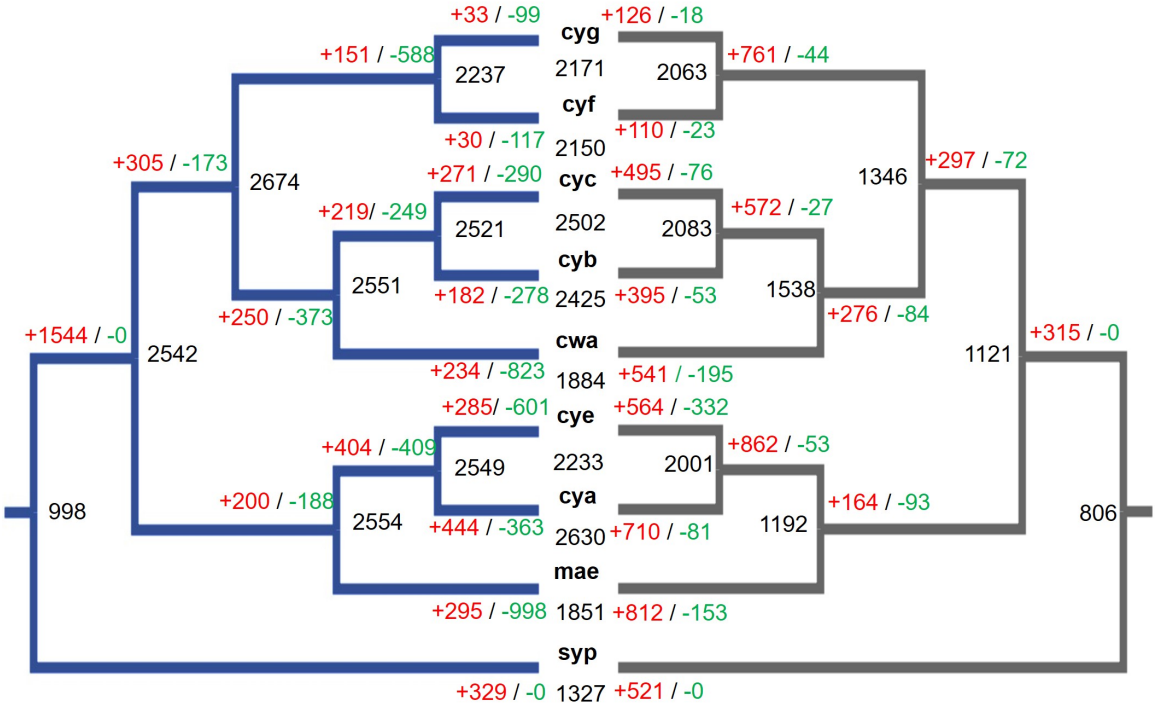



Figure 7.1: Ancestral genome content inferred using Notung 2.8, with Wagner parsimony for comparison. The number of inferred ancestral families shown on nodes. The number of inferred gains and losses indicated by +/- on branches. Species abbreviations: *cwa*: *Crocosphaera watsonii* WH 8501; *cya*: *Cyanothece* sp. PCC 7424; *cyb*: *Cyanothece* sp. ATCC 51142; *cyc*: *Cyanothece* sp. CCY 0110; *cyd*: *Cyanothece* sp. PCC 7425; *cye*: *Cyanothece* sp. PCC 7822; *cyf*: *Cyanothece* sp. PCC 8801; *cyg*: *Cyanothece* sp. PCC 8802; *mae*: *Microcystis aeruginosa* NIES843; *syp*: *Synechocystis* sp. PCC6803.

The heatmap in Figure 7.2 summarizes inferred horizontal gene transfers in the Cyanothece distributed genome. These results show that HGT activity varies substantially across strains. For example, *cya* is both a donor and a recipient of genetic material, while *mae* is a preferential donor of HGT.

The ancestral gene content shown in Figure 7.1 can be used to identify interesting gene family gain and loss between specific divergence of species and connect the gene events with the ecological changes. For example, The *Microcystis aeruginosa* NIES 843 (*mae*) causes toxic blooms in freshwater habitats. NOTUNG 2.9 inferred 1000 gene family losses in the lineage leading to *mae*. We explored the ecological implications of this genome reduction using the enrichKegg function in the Bioconductor clusterProfiler package (Yu et al., 2012). Interestingly, significant losses in *mae* include genes encoding the phycoerythrin proteins found in phycobilisomes, which harvest light for photosynthesis. Phycoerythrins harvest

green light, found at lower levels in the water column. However, *mae* has gas filled vesicles that allow it to live at the surface of the water column, where wavelengths at the red end of the spectrum are more prevalent. This is consistent with the hypothesis that *mae* lost its phycoerythrin genes in the adaptation to its water-surface niche.

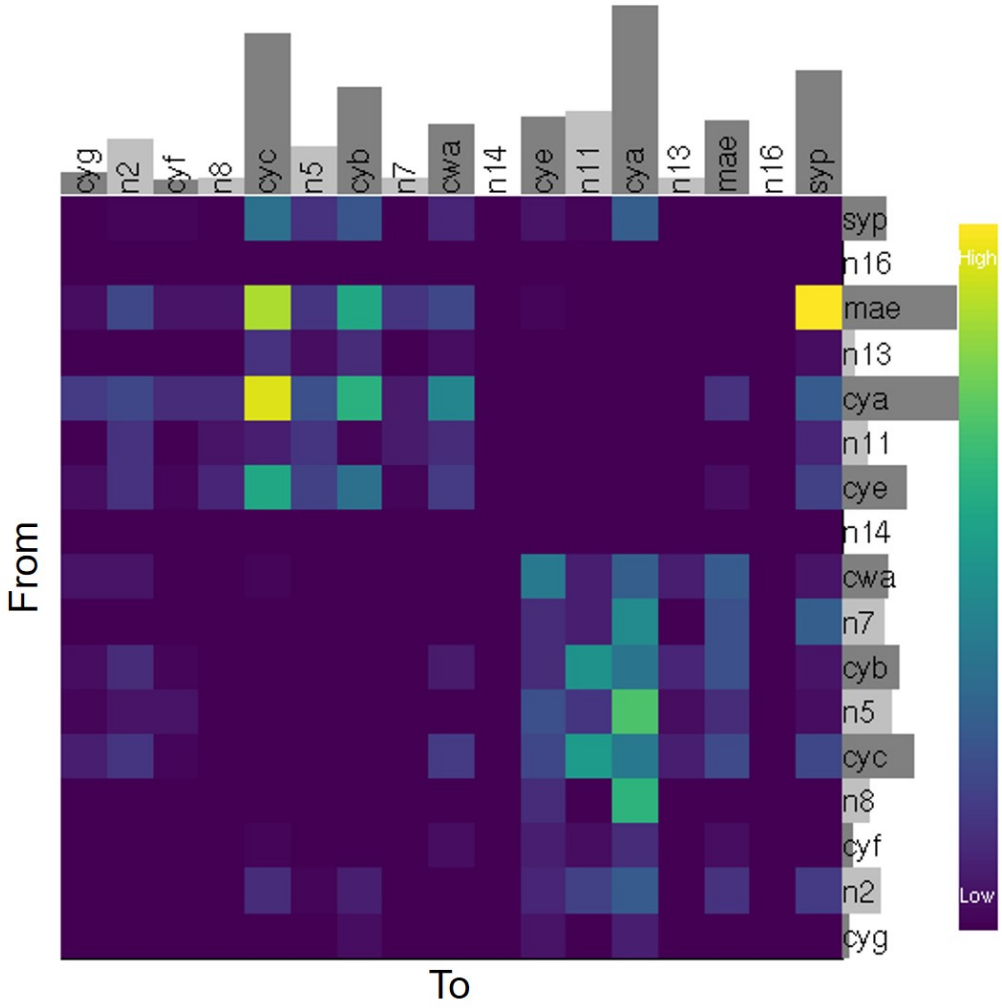



Figure 7.2: Heatmap showing the number of genes horizontally transferred between all pairs of species in our data set. Barplots on top and right shows the number of genes transferred from/received by each species. Calculated in R from tables output by NOTUNG 2.9.

# Chapter 8

# Discussion

Reconciliation between gene tree and species tree has been one of the major ways to understand history of gene families among species. Inferring gene events allows researchers to study various biological questions, such as the correlation between gene family expansion and species evolution and gene function evolution with development of metabolic processes, and the co-evolution between gene family and the ecological niche. With the advancement of sequencing technology, genome scale phylogenetic data sets of gene trees can be constructed and reconciliation can be applied to thousands of gene trees in an automated fashion. Earlier models for reconciliation aimed to explain the gene tree incongruence with duplication and loss event only. Increasing evidence that horizontal gene transfer (HGT), duplication, and loss all play a role in gene family evolution suggests that a reconciliation model that infers all three gene events together is urgently needed.

However, incorporating transfer events in the reconciliation model introduces several challenges:

1. HGT can occur between any two species that are not related by vertical descent. This allows the genes to jump in many places in the species tree. This species feature of HGT allows a given instance of gene tree incongruence to be explained by multiple event histories. Generating all event histories for exploratory analysis for a gene tree is algorithmically more complex than generating a single optimal history. . To separate these solutions, a cost vector $< C_T, C_D, C_L >$ is needed to prefer some event histories from others. For each event history, the cost is calculated as the weighted sum of

| Gene Tree vs Species Tree | Binary $T_S$ | Non-binary $T_S$ |
|---|---|---|
| Binary $T_G$ | DTL | DTL(Ch.3) |
| Non-binary $T_G$ | DTL(Ch.4) | NA |

Table 8.1: The algorithms that exist for the input gene tree and species tree under different event models. Red models indicates my contribution.

the event numbers and the event costs. Under maximum parsimony framework, the solution with the lowest cost is considered the best solution.

2. HGT also introduces temporal constraints. The donor species and recipient species must exist at the same time for HGT to occur. Therefore, the event history must be temporally feasible to be a valid solution. The problem of finding the lowest cost solution that is temporally feasible is NP-complete (Hallett et al., 2004).

My thesis focusses on several of the challenges with the DTL event model. As Table 8.1 shows, under DTL model, the input gene tree and species tree can be either binary or non-binary. Many algorithmic studies have been done on DTL reconciliation of a binary gene tree with a binary species tree. Less work has been completed on the DTL-reconciliation of a binary gene tree with a non-binary species tree, or a non-binary gene tree with a binary species tree. In my thesis, I have described several of my contributions in all three problems.

## 8.1. DTL Reconciliation of Binary Gene Tree and Binary Species Tree

Although many studies have been done on parsimony based reconciliation of binary gene tree with an undated binary species tree, our algorithm and implementation of the algorithm was the first one to report all optimal solutions (Stolzer et al., 2012b) and is still one of the only two algorithms that does this (Donati et al., 2015a). This allows an exploratory analysis for user to investigate each solutions manually for a gene tree. However, some gene trees may result in too many optimal solutions, making manual inspection obsolete. A way of summarizing all optimal solutions for each gene tree reconciliation automatically would provide a faster understanding of the evolution of the gene family for the user.

The current approach to DTL reconciliation is to find all candidate solutions with the minimum cost using dynamic programming. If a candidate solution is temporally feasible, then it is also an optimal solution. However, if all solutions with the minimum cost are temporally infeasible, the only known approach is to through exhaustive search with super exponential complexity. This leaves a exciting challenge for the future study. Between the dynamic programming (Section 2.3.2) and the exponential complexity search, there may exist a middle ground for finding pseudo optimal solutions when all minimum cost candidate solutions are temporally infeasible.

I also discussed a model for reconciling binary tree on three levels of biological organization. The model describes a single domain evolving through domain shuffling events in gene families and the gene tree evolving through gene events in species tree (Section 3.3). I define the domain shuffling events to the domain transfer, domain duplication, domain insertion, and domain loss. Under this model, the domain family's history can be represented by a domain tree; and the genes, where these domains are sampled from, can be represented by a gene tree if they are from the same gene family. The co-evolution between the domain family and gene family can be studied by inferring the domain shuffling events by comparing the two trees. In the context of tree reconciliation, I extended the DTL reconciliation algorithm for gene tree to infer domain duplication, transfer, insertion, and loss event for domain families.

## 8.2. DTL reconciliation of A Binary Gene Tree and A Non-binary Species Tree

In chapter 3, I discussed an approach to represent short species tree branch, where incomplete lineage sorting (ILS) is likely to occur. To avoid the incorrect inference of gene events, short species tree branches can be collapsed, resulting in a non-binary species tree. I also described a reconciliation algorithm with a heuristic approach for distinguishing gene tree incongruence caused by gene events from incongruent branching patterns that are consistent with ILS. Under the DTL event model, this algorithm takes an input of a binary gene tree, and a non-binary species tree, and infers gene duplication, transfer, and loss events (Table 8.1). Incongruent gene tree nodes that may reflect incongruent co-divergence with speciation are also inferred.

Our model for non-binary species tree reconciliation, is a compromise between the existing DTL reconciliation model, and the coalescent model, which estimates the probability of gene tree incongruence due to incomplete lineage sorting, as a function of population size and the time between speciations. These population parameters are usually unknown. Our parsimony model, on the other hand, does not take branch lengths into account. Instead, the model collapses the short branches into a polytomy and does not infer gene events when the incongruence that can be explained by ILS. This potentially introduces type II error, where the gene events are underestimated. This conservative approach aims to reduce the type I error. In future work, simulation studies could be used to characterize the accuracy of this approach and measure the over-estimation if using binary species tree and the under-estimation if using non-binary species tree.

## 8.3. DTL Reconciliation of a Non-binary gene tree and a Binary Species Tree

In Chapter 4, I discussed the scenario where the gene tree contains uncertain branches, which the sequence information lacked enough signal to infer a strongly supported branching order. This uncertainty can result in a non-binary gene tree. The exact algorithm for reconciliation of a non-binary gene trees with a binary species tree under the DTL model has exponential complexity. As a result, the exact algorithm becomes intractable with increasing polytomy size. I have proposed heuristics, which can be used as stand-alone tree correction methods or as subroutines in species tree-aware gene tree construction techniques.

I demonstrated the effectiveness of these heuristics experimentally by comparing the event costs obtained with each heuristic, with the minimum event cost obtained with the exact method. I analyzed the effect of different number of iterations when applying *Hybrid* and *NNI* heuristic. The result suggests that *Local DL* provides a good starting resolution, greatly improving the accuracy of the *Hybrid* approach.

A hill climbing NNI algorithm, developed by Nguyen et al. (2013) was originally applied to Resolve-DTL problem specifically for dated species tree. However, the algorithm can be applied to undated species tree with slight modifications. The comparison between the heuristics in this study and the hill climbing NNI would be an interesting topic for future studies.

The NNI approach focuses on switching the neighbours that are adjacent to each other in the tree. NNI may be to fine grained when searching for resolutions with horizontal gene transfers which can introduce big changes in the topology. A different technique, Subtree Prune and Regrafting (SPR) is another strategy to search the tree space. SPR, which makes larger jumps in the tree space, would also be a good future direction.

I also calculated the Robinson-Foulds (RF) distance between the rearranged trees obtained with the heuristic and exact methods. The RF distance is frequently used to compare trees obtained with different methods for the same input data. Zheng and Zhang (2014b) have compared event-based distances with RF distances and have shown that several metrics based on the DL model are not equivalent to RF distance. Further, they observed that event-based costs are "more sensitive to topology" than RF distances. Our empirical observations suggest that a measure based on DTL-events is similarly more sensitive to topology. The relationship between DTL- event metrics and RF distance may be a fruitful direction for future theoretical studies. In the context of gene tree-species tree reconciliation, trees that minimize the RF distance may not give a satisfying low event cost (Zheng and Zhang, 2014b)

This class of algorithms for the Resolve-D(T)L problem focused on the resolving the gene tree uncertainty has an important limitation that should be given greater attention. Such methods typically assume that all incongruence is due to either gene events or phylogenetic error. However, population process (such as ILS, migration, hybridization, and introgression) introduce uncertainty that may propagate to the gene tree. The resulting incongruence may be the key signature for inferring the population processes, which could be removed when an uncertain branch is resolved by a species-tree-aware method. Thus, if the uncertanty of the gene tree is due to population processes, the reconciliation-based, species tree aware method could introduce error, rather than decreasing it.

## 8.4. Event Cost Selection

One of the major challenges of using the DTL reconciliation algorithms with a parsimony criterion is that the algorithm requires a pre-set event cost vector, which is used to calculate the cost of reconciliation result. Which cost to select in order to obtain the most accurate result has always been an important question. In chapter 5 and 6, I approached cost selection from two directions. In chapter 5, under different assumptions concerning the event rates

and the structure of the gene tree, I try to identify a range of event rates where a selected cost vector can be chosen such taht the maximum parsimony reconciliations may be equivalent to the most likely reconciliation.

In chapter 5, I constructed four log likelihood functions under four different scenarios and compared the maximum likelihood with the maximum parsimony optimization criteria. In birth-death lineage model, the log likelihood function in a linear function of $N_D, N_T, N_L$ which fits very well to maximum parsimony. For constant lineage and birth-death number models, there is an additional term that indicates some level of interaction between gene event rates. For constant number model, the log likelihood can be separated into independent functions of $N_D, N_T$, and $N_L$, however, these are not linear. For these three models, it is not clear whether there exists a cost vector, such that MP and ML are equivalent. Further analysis on the interaction between the three event types to categorize their independence and find the conditions for which the three events can be treated as three independent functions would allow more understanding to the equivalence between maximum likelihood and maximum parsimony reconciliation models.

The theoretical work in the chapter left several exciting challenges to be solved, including the use of a probability generating function to solve a partial differential equation. The solution obtained so far is an arbitrary function that needs to be narrowed down by additional constraints. Further investigation on such constraints and derivation of probability distribution of the number of birth and death events would significantly advance the comparison between maximum likelihood and maximum parsimony.

The factor that describes the number of duplicate topologies when the probability of a reconciliation is calculated as a product of probability of every gene subtree embedded in species tree branch. This term is another challenge that is not well understood in terms of its variance and other properties. If this term could be neglected in the maximum likelihood optimization criteria under some condition, it would also advance the comparison of the maximum likelihood and maximum parsimony optimization criteria.

Because of the fundamental difference between MP and ML optimization criterion of reconciliation, I also approach the problem of cost selection from another perspective. Given a gene tree and a species tree, the space of all cost vectors can be partitioned into regions, in which all cost vectors in the same region will give the same result for a given pair of gene and species trees. In phylogenomic analysis with thousands of gene trees, generating such a par-

tition for each gene tree is not feasible. Further analysing the cost space indicates that some partitions may commonly exist in many gene trees. This suggests that a partition of the cost space that captures these common partitions will greatly facilitate the cost selection, because the cost vectors from different common partitions are likely to generate different reconciliation results. Analysing and comparing these different results can provide more insights into the structure of all the gene trees. I described several common trade-offs and demonstrated the gene tree scenarios that these common trade-offs represent. This provides a basis for users to combine their biological intuition with the selection of cost vectors.

# Appendix

## Derivation of $P_{B,D}(t)$

The equation to estimate the probability distribution of the number of birth and death event after time $t$ in Chapter 5 (Eq. 5.36) is derived here.

$$\begin{aligned}\frac{dP_{N_B,N_L}(t)}{dt} =& P_{N_B-1,N_L}(t)(N_B - N_L)r_B \\ &+ P_{N_B,N_L-1}(t)(N_B - N_L + 2)r_L \\ &- P_{N_B,N_L}(t)(1 + N_B - N_L)(r_B + r_L)\end{aligned} \tag{A1}$$

In order to solve this differential equation, we introduce $\phi(N_B, N_L, t)$, a probability generating function (PGF) with two variables for the probability $P_{N_B,N_L}(t)$:

$$\phi(N_B, N_L, t) = \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} N_B^i N_L^j P_{i,j}(t). \tag{A2}$$

For simplicity, I use $x$ and $y$ to represent the $N_B$ and $N_L$ in the above differential equation.

$$\phi(x, y, t) = \sum_{i,j} x^i y^j P_{i,j}(t). \tag{A3}$$

Taking the derivative with respect to $t$ on both side gives:

$$\frac{\partial \phi}{\partial t} = \sum_{i,j} x^i y^j \left( (i-j)\lambda \frac{dP_{i-1,j}(t)}{dt} - (1+i-j)(\lambda+\mu)\frac{dP_{i,j}(t)}{dt} + (i-j+2)\mu \frac{dP_{i,j-1}(t)}{dt} \right) \quad \text{(A4)}$$

$$= [\lambda(x-1) - \mu(y-1)](\sum_{i,j}(i-j+1)x^i y^j P_{i,j}(t)) \quad \text{(A5)}$$

$$= [\lambda(x-1) - \mu(y-1)](x\frac{\partial \phi}{\partial x} - y\frac{\partial \phi}{\partial y} + \phi) \quad \text{(A6)}$$

Let $\beta = [\lambda(x-1) - \mu(y-1)]$, $\phi_t = \frac{d\phi}{dt}$, $x_t$‘$\frac{dx}{dt}$, $y_t$‘$\frac{dy}{dt}$. Then, the auxiliary equation of the PDE is:

$$\frac{dt}{1} = \frac{dx}{-x\beta} = \frac{dy}{y\beta} = \frac{d\phi}{\beta\phi}, \quad \text{(A7)}$$

which gives a series of ordinary differential equations:

$$\frac{dx}{dt} = -x\beta, \quad \text{(A8)}$$

$$\frac{dy}{dt} = y\beta, \quad \text{(A9)}$$

$$\frac{dy}{dx} = -\frac{y}{x}, \quad \text{(A10)}$$

$$\frac{d\phi}{dx} = -\frac{\phi}{x}. \quad \text{(A11)}$$

Reordering Equations A8- A11 and taking the derivative with respect to $t$ results in:

$$y = \left[\frac{1}{x}\frac{dx}{dt} + \lambda(x-1)\right]\frac{1}{\mu} + 1, \quad \text{(A12)}$$

$$\frac{dy}{dt} = \frac{1}{\mu}\left[-\frac{1}{x^2}\left(\frac{dx}{dt}\right)^2 + \frac{1}{x}\frac{d^2x}{dt^2} + \lambda\frac{dx}{dt}\right]. \quad \text{(A13)}$$

Substituting Eqn. A12 and A13 in Eqn. A9, we obtain

$$\frac{1}{\mu}\left[-\frac{1}{x^2}\left(\frac{dx}{dt}\right)^2 + \frac{1}{x}\frac{d^2x}{dt^2} + \lambda\frac{dx}{dt}\right] = \left[\left[\frac{1}{x}\frac{dx}{dt} + \lambda(x-1)\right]\frac{1}{\mu} + 1\right]\left(\frac{1}{-x}\frac{dx}{dt}\right) \quad \text{(A14)}$$

The above equation can be simplified to a second order ODE:

$$\frac{d^2x}{dt^2} + (2\lambda x + \mu - \lambda)\frac{dx}{dt} = 0. \quad \text{(A15)}$$

Let $v = \frac{dx}{dt}$, then $\frac{d^2x}{dt^2} = \frac{dv}{dt} = \frac{dx}{dt}\frac{dv}{dx} = v\frac{dv}{dx}$. Therefore,

$$v\frac{dv}{dx} + (2\lambda x + \mu - \lambda)v = 0. \tag{A16}$$

Integrating $\frac{dv}{dx}$, we get

$$v = \lambda x^2 + (\mu - \lambda)x + C = \frac{dx}{dt}. \tag{A17}$$

For simplicity here, we assume $C = 0$ in Eq. A17. Integrating $\frac{dx}{dt}$ yields,

$$\frac{xe^{(\lambda-\mu)t}}{\lambda x + u - \lambda} = C_1. \tag{A18}$$

Solving Eq. A10 and Eq. A11, we obtain

$$xy = C_2 \tag{A19}$$

$$x\phi = C_3 \tag{A20}$$

$$\tag{A21}$$

Let $C_3 = \Phi(C_1, C_2)$, where $\Phi$ is an arbitrary function, the general solution is :

$$\phi(x, y, t) = \frac{1}{x}\Phi(xy, \frac{xe^{(\lambda-\mu)t}}{\lambda x + \mu - \lambda}) \tag{A22}$$

To narrow down the solution space for this differential equation, more restrictions are required. Lacking these restrictions, solving this set of ODEs is intractable.

# Bibliography

Orjan Akerborg, Bengt Sennblad, Lars Arvestad, and Jens Lagergren. Simultaneous Bayesian gene tree reconstruction and reconciliation analysis. *PNAS*, 106:5714–5719, Apr 2009.

Ricard Albalat and Cristian Cañestro. Evolution by gene loss. *Nat Rev Genet*, 17:379–391, Jul 2016.

Cheryl P. Andam and J. Peter Gogarten. Biased gene transfer in microbial evolution. *Nat Rev Microbiol*, 9:543–555, Jul 2011.

Jan O. Andersson. Horizontal gene transfer between microbial eukaryotes. *Methods Mol Biol*, 532:473–487, 2009.

Maria Anisimova, Manuel Gil, Jean-François Dufayard, Christophe Dessimoz, and Olivier Gascuel. Survey of branch support methods demonstrates accuracy, power, and robustness of fast likelihood-based approximation schemes. *Syst Biol*, 60(5):685–699, Oct 2011.

Lars Arvestad, Ann-Charlotte Berglund, Jens Lagergren, and Bengt Sennblad. Bayesian gene/species tree reconciliation and orthology analysis using MCMC. *Bioinformatics*, 19 Suppl 1:i7–15, 2003.

Lars Arvestad, Jens Lagergren, and Bengt Sennblad. The gene evolution model and computing its associated probabilities. *J. ACM*, 56(2):1–44, 2009.

J. C. Avise, J. F. Shapira, S. W. Daniel, C. F. Aquadro, and R. A. Lansman. Mitochondrial DNA differentiation during the speciation process in *Peromyscus*. *Mol Biol Evol*, 1(1): 38–56, Dec 1983.

J. C. Avise, J. E. Neigel, and J. Arnold. Demographic influences on mitochondrial dna lineage survivorship in animal populations. *J Mol Evol*, 20(2):99–105, 1984.

Rajeev K. Azad and Jeffrey G. Lawrence. Detecting laterally transferred genes. *Methods Mol Biol*, 855:281–308, 2012.

Mukul S. Bansal, Eric J. Alm, and Manolis Kellis. Efficient algorithms for the reconciliation problem with gene duplication, horizontal transfer and loss. *Bioinformatics*, 28:i283–i291, Jun 2012.

Mukul S. Bansal, Eric J. Alm, and Manolis Kellis. Reconciliation revisited: Handling multiple optima when reconciling with duplication, transfer, and loss. *J Comput Biol*, Sep 2013.

Mukul S. Bansal, Yi-Chieh Wu, Eric J. Alm, and Manolis Kellis. Improved gene tree error correction in the presence of horizontal gene transfer. *Bioinformatics*, 31:1211–1218, Apr 2015.

Daniel Barker. Gene trees for orthologous groups from: The evolution of nitrogen fixation in cyanobacteria, 2013. Edinburgh DataShare. http://dx.doi.org/10.5061/dryad.pv6df.

Alix Boc, Hervé Philippe, and Vladimir Makarenkov. Inferring and validating horizontal gene transfer events using bipartition dissimilarity. *Syst Biol*, 59:195–211, Mar 2010.

Paola Bonizzoni, Gianluca Della Vedova, and Riccardo Dondi. Reconciling a gene tree to a species tree under the duplication cost model. *Theor. Comput. Sci.*, pages 36–53, 2005.

Bastien Boussau, Gergely J. Szöllosi, Laurent Duret, Manolo Gouy, Eric Tannier, and Vincent Daubin. Genome-scale coestimation of species and gene trees. *Genome Res*, 23: 323–330, Feb 2013.

Thomas Brockmöller, Zhihao Ling, Dapeng Li, Emmanuel Gaquerel, Ian T. Baldwin, and Shuqing Xu. *Nicotiana attenuata* Data Hub (NaDH): an integrative platform for exploring genomic, transcriptomic and metabolomic data in wild tobacco. *BMC Genomics*, 18:79, Jan 2017.

M. A. Charleston. Jungles: a new solution to the host/parasite phylogeny reconciliation problem. *Math Biosci*, 149:191–223, May 1998.

Ruchi Chaudhary, J. Gordon Burleigh, and Oliver Eulenstein. Efficient error correction algorithms for gene tree reconciliation based on duplication, duplication and loss, and deep coalescence. *BMC Bioinformatics*, 13 Suppl 10:S11, Jun 2012.

Cedric Chauve, Jean-Philippe Doyon, and Nadia El-Mabrouk. Gene family evolution by duplication, speciation, and loss. *J Comput Biol*, 15:1043–1062, Oct 2008.

Cedric Chauve, Nadia El-Mabrouk, Laurent Guéguen, Magali Semeria, and Eric Tannier. Duplication, rearrangement and reconciliation: A follow-up 13 years later. volume 19 of *Computational Biology*, pages 47–62. Springer London, 2013.

Kevin Chen, Dannie Durand, and Martin Farach-Colton. Notung: Dating gene duplications using gene family trees. In *RECOMB 2000: Proceedings of the Fourth International Conference on Research in Computational Biology*, ACM Press, pages 96–106, New York, NY, USA, 2000. ACM Press.

Chris Conow, Daniel Fielder, Yaniv Ovadia, and Ran Libeskind-Hadas. Jane: a new tool for the cophylogeny reconstruction problem. *Algorithms Mol Biol*, 5:16, Feb 2010.

T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. MIT Press/McGraw-Hill, 1990.

Miklós Csurös. Count: evolutionary analysis of phylogenetic profiles with parsimony and likelihood. *Bioinformatics*, 26:1910–1912, Aug 2010.

Daniel A. Dalquen, Maria Anisimova, Gaston H. Gonnet, and Christophe Dessimoz. Alf-simulation framework for genome evolution. *Molecular Biology and Evolution*, 29(4):1115, 2011.

Charlotte A. Darby, Maureen Stolzer, Patrick J. Ropp, Daniel Barker, and Dannie Durand. Xenolog classification. *Bioinformatics*, 33(5):640–649, Mar 2017.

Lawrence A. David and Eric J. Alm. Rapid evolutionary innovation during an Archaean genetic expansion. *Nature*, 469:93–96, Jan 2011.

B. Donati, C. Baudet, B. Sinaimeri, P. Crescenzi, and M. F. Sagot. EUCALYPT: efficient tree reconciliation enumerator. *Algorithms Mol Biol*, 10(1):3, 2015a.

Beatrice Donati, Christian Baudet, Blerina Sinaimeri, Pierluigi Crescenzi, and Marie-France Sagot. Eucalypt: efficient tree reconciliation enumerator. *Algorithms Mol Biol*, 10:3, Jan 2015b.

J. Doyon and C. Chauve. Branch-and-bound approach for parsimonious inference of a species tree from a set of gene family trees. *Adv Exp Med Biol*, 696:287–295, 2011.

J. Doyon, C. Scornavacca, K. Gorbunov, G. Szollosi, V. Ranwez, and V. Berry. An efficient algorithm for gene/species trees parsimonious reconciliation with losses, duplications and transfers. In Eric Tannier, editor, *RECOMB-CG 2010: Proceedings of the International Workshop on Comparative Genomics*, volume 6398 of *LNBI*, pages 93–108. Springer, 2010.

Jean-Philippe Doyon, Vincent Ranwez, Vincent Daubin, and Vincent Berry. Models, algorithms and programs for phylogeny reconciliation. *Brief Bioinform*, 12:392–400, Sep 2011.

Irina S. Druzhinina, Eva M. Kubicek, and Christian P. Kubicek. Several steps of lateral gene transfer followed by events of 'birth-and-death' evolution shaped a fungal sorbicillinoid biosynthetic gene cluster. *BMC Evolutionary Biology*, 16(1):269, Dec 2016.

Julie C. Dunning Hotopp. Horizontal gene transfer between bacteria and animals. *Trends Genet*, 27(4):157–163, Apr 2011.

D. Durand and R. A. Hoberman. Diagnosing duplications: can it be done? *Trends Genet*, 22(3):156–64, Mar 2006.

D. Durand, B. V. Halldorsson, and B. Vernot. A hybrid micro-macroevolutionary approach to gene tree reconstruction. In *Proceedings of the Ninth Annual International Conference on Computational Molecular Biology (RECOMB)*, 2005.

D. Durand, B. Halldorsson, and B. Vernot. A hybrid micro-macroevolutionary approach to gene tree reconstruction. *Journal of Computational Biology*, 13(2):320–335, 2006a. A preliminary version appeared in Recomb 2005, LNBI 3500, Springer Verlag, 250-264.

D. Durand, B. Halldorsson, and B. Vernot. A hybrid micro-macroevolutionary approach to gene tree reconstruction. *J Comput Biol*, 13(2):320–335, 2006b. A preliminary version appeared in RECOMB 2005, LNBI 3500, Springer Verlag, 250–264.

O. Eulenstein and M. Vingron. On the equivalence of two tree mapping measures. *Discrete Applied Mathematics*, 88(13):103 – 128, 1998. Computational Molecular Biology {DAM} - {CMB} Series.

James S. Farris. A probability model for inferring evolutionary trees. *Systematic Zoology*, 22(3):250–256, 1973.

Joseph Felsenstein. Maximum likelihood and minimum-steps methods for estimating evolutionary trees from data on discrete characters. *Systematic Biology*, 22(3):240, 1973.

Joseph Felsenstein. *Inferring Phylogenies*. Sinauer, 2003.

David A. Fitzpatrick. Horizontal gene transfer in fungi. *FEMS Microbiol Lett*, 329:1–8, Apr 2012.

Felipe J. Fuzita, Martijn W. H. Pinkse, José S. L. Patane, Peter D. E. M. Verhaert, and Adriana R. Lopes. High throughput techniques to reveal the molecular physiology and evolution of digestion in spiders. *BMC Genomics*, 17:716, Sep 2016.

Caihua Gao, Xiaodong Ren, Annaliese S. Mason, Honglei Liu, Meili Xiao, Jiana Li, and Donghui Fu. Horizontal gene transfer in plants. *Funct Integr Genomics*, 14(1):23–29, Mar 2014.

W. Gilbert. Why genes in pieces? *Nature*, 271(5645):501, Feb 1978.

W. Gilbert. The exon theory of genes. *Cold Spring Harb. Symp. Quant. Biol.*, 52:901–905, 1987.

David A. Gold, Jonathan Grabenstatter, Alex de Mendoza, Ana Riesgo, Iñaki Ruiz-Trillo, and Roger E. Summons. Sterol and genomic analyses validate the sponge biomarker hypothesis. *Proceedings of the National Academy of Sciences*, 113(10):2684–2689, 2016.

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

Pawel Gorecki and Oliver Eulenstein. Algorithms: simultaneous error-correction and rooting for gene tree reconciliation and the gene duplication problem. *BMC Bioinformatics*, 13 Suppl 10:S14, Jun 2012.

Pawel Górecki and Oliver Eulenstein. DrML: probabilistic modeling of gene duplications. *J Comput Biol*, 21:89–98, Jan 2014.

Stéphane Guindon, Jean-François Dufayard, Vincent Lefort, Maria Anisimova, Wim Hordijk, and Olivier Gascuel. New algorithms and methods to estimate maximum-likelihood phylogenies: assessing the performance of PhyML 3.0. *Syst Biol*, 59:307–321, May 2010.

Mike Hallett, Jens Lagergren, and Ali Tofigh. Simultaneous identification of duplications and lateral transfers. In *RECOMB 2004: Proceedings of the Eigth International Conference*

*on Research in Computational Biology*, ACM Press, pages 347–356, New York, NY, USA, 2004.

E. F. Harding. The probabilities of rooted tree-shapes generated by random bifurcation. *Advances in Applied Probability*, 3(1):44–77, 1971.

Paul V. Hickner, Chissa L. Rivaldi, Cole M. Johnson, Madhura Siddappaji, Gregory J. Raster, and Zainulabeuddin Syed. The making of a pest: Insights from the evolution of chemosensory receptor families in a pestiferous and invasive fly, *Drosophila suzukii*. *BMC Genomics*, 17:648, Aug 2016.

Tobias Hill, Karl J. V. Nordström, Mikael Thollesson, Tommy M. Säfström, Andreas K. E. Vernersson, Robert Fredriksson, and Helgi B. Schiöth. Sprit: Identifying horizontal gene transfer in rooted phylogenetic trees. *BMC Evol Biol*, 10:42, Feb 2010.

J. P. Huelsenbeck, B. Rannala, and B. Larget. A Bayesian framework for the analysis of cospeciation. *Evolution*, 54:352–364, Apr 2000.

Daniel H. Huson and Celine Scornavacca. A survey of combinatorial methods for phylogenetic networks. *Genome Biol Evol*, 3:23–35, Nov 2011.

Jun Inoue, Yukuto Sato, Robert Sinclair, Katsumi Tsukamoto, and Mutsumi Nishida. Rapid genome reshaping by multiple-gene loss after whole-genome duplication in teleost fish suggested by mathematical modeling. *Proceedings of the National Academy of Sciences*, 112(48):14918–14923, 2015.

Edwin Jacox, Cedric Chauve, Gergely J. Szöll?si, Yann Ponty, and Celine Scornavacca. ecceTERA: comprehensive gene tree-species tree reconciliation using parsimony. *Bioinformatics*, 32:2056–2058, Jul 2016.

Edwin Jacox, Mathias Weller, Eric Tannier, and Celine Scornavacca. Resolution and reconciliation of non-binary gene trees with transfers, duplications and losses. *Bioinformatics*, 33:980–987, Apr 2017.

R. Jain, M. C. Rivera, and J. A. Lake. Horizontal gene transfer among genomes: the complexity hypothesis. *Proc Natl Acad Sci U S A*, 96(7):3801–3806, Mar 1999.

Henrik Kaessmann, Sebastian Zollner, Anton Nekrutenko, and Wen-Hsiung Li. Signatures of domain shuffling in the human genome. *Genome Res*, 12(11):1642–1650, Nov 2002.

Thomas M. Keane, Christopher J. Creevey, Melissa M. Pentony, Thomas J. Naughton, and James O. Mclnerney. Assessment of methods for amino acid matrix selection and their use on empirical data shows that ad hoc assumptions for choice of matrix are not justified. *BMC Evol Biol*, 6:29, Mar 2006.

David G. Kendall. Stochastic processes and population growth. *J. of the Royal Statistial Society Series B*, 11(2):230–282, 1949.

Mehmood Alam Khan, Owais Mahmudi, Ikram Ullah, Lars Arvestad, and Jens Lagergren. Probabilistic inference of lateral gene transfer events. *BMC Bioinformatics*, 17(14):431, 2016.

Jung-Woong Kim, Hyun-Jin Yang, Adam Phillip Oel, Matthew John Brooks, Li Jia, David Charles Plachetzki, Wei Li, William Ted Allison, and Anand Swaroop. Recruitment of rod photoreceptors from short-wavelength-sensitive cones during the evolution of nocturnal vision in mammals. *Dev Cell*, 37:520–532, Jun 2016a.

Ki-Tae Kim, Jongbum Jeon, Jaeyoung Choi, Kyeongchae Cheong, Hyeunjeong Song, Gobong Choi, Seogchan Kang, and Yong-Hwan Lee. Kingdom-wide analysis of fungal small secreted proteins (ssps) reveals their potential role in host association. *Front Plant Sci*, 7: 186, Feb 2016b.

Konstantinos T. Konstantinidis, Alban Ramette, and James M. Tiedje. The bacterial species definition in the genomic era. *Proceedings of the Royal Society B: Biological Sciences*, 361 (1475):1929–1940, 11 2006.

M Kordi and S Bansal. On the complexity of duplication-transfer-loss reconciliation with non-binary gene trees. *IEEE/ACM Trans Comput Biol Bioinform*, 14(3):587–599, May/Jun 2017.

Misagh Kordi and Mukul S. Bansal. Exact algorithms for duplication-transfer-loss reconciliation with non-binary gene trees. In *ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, pages 297–306, 2016.

Sanna Koskiniemi, Song Sun, Otto G. Berg, and Dan I. Andersson. Selection-driven gene loss in bacteria. *PLOS Genetics*, 8(6):1–7, 06 2012.

Manuel Lafond, Magali Semeria, Krister M. Swenson, Eric Tannier, and Nadia El-Mabrouk. Gene tree correction guided by orthology. *BMC Bioinformatics*, 14 Suppl 15:S5, Oct 2013.

Manuel Lafond, Cedric Chauve, Riccardo Dondi, and Nadia El-Mabrouk. Polytomy refinement for the correction of dubious duplications in gene trees. *Bioinformatics*, 30:i519–i526, Sep 2014.

Manuel Lafond, Riccardo Dondi, and Nadia El-Mabrouk. The link between orthology relations and gene trees: a correction perspective. *Algorithms Mol Biol*, 11:4, Apr 2016.

H. Lai, M. Stolzer, and D. Durand. Fast heuristics for resolving weakly supported branches using duplication, transfers, and losses. In J. Meidanis and L. Nakhleh, editors, *15th RECOMB Satellite Conference on Comparative Genomics*, Lecture Notes in Bioinformatics. Springer Verlag, 2017. in press.

N. Latysheva, V. L. Junker, W. J. Palmer, G. A. Codd, and D. Barker. The evolution of nitrogen fixation in cyanobacteria. *Bioinformatics*, 28(5):603–6, 2012.

J. G. Lawrence. Gene transfer, speciation, and the evolution of bacterial genomes. *Curr Opin Microbiol*, 2:519–523, Oct 1999.

Ran Libeskind-Hadas and Michael A. Charleston. On the computational complexity of the reticulate cophylogeny reconstruction problem. *J Comput Biol*, 16:105–117, Jan 2009.

Ran Libeskind-Hadas, Yi-Chieh Wu, Mukul S. Bansal, and Manolis Kellis. Pareto-optimal phylogenetic tree reconciliation. *Bioinformatics*, 30:i87–i95, Jun 2014.

M. Long, E. Betran, K. Thornton, and W. Wang. The origin of new genes: glimpses from the young and old. *Nat Rev Genet*, 4(11):865–75, Nov 2003.

Lafond M., Swenson K.M., and El-Mabrouk N. An optimal reconciliation algorithm for gene trees with polytomies. In Raphael B. and Tang J., editors, *Algorithms in Bioinformatics*, volume 7534 of *LNCS*, pages 106–122. Springer, 2012.

W. P. Maddison. Gene trees in species trees. *Syst. Biol.*, 46(3):523–536, 1997.

Santoshkumar Magadum, Urbi Banerjee, Priyadharshini Murugan, Doddabhimappa Gangapur, and Rajasekar Ravikesavan. Gene duplication as a major force in evolution. *J Genet*, 92:155–161, Apr 2013.

Sean K. McKenzie, Ingrid Fetter-Pruneda, Vanessa Ruta, and Daniel J. C. Kronauer. Transcriptomics and neuroanatomy of the clonal raider ant implicate an expanded clade of

odorant receptors in chemical communication. *Proc Natl Acad Sci U S A*, 113:14091–14096, Dec 2016.

Olivia Mendivil Ramos and David E. K. Ferrier. Mechanisms of gene duplication and translocation and progress towards understanding their relative contributions to animal genome evolution. *Int J Evol Biol*, 2012:846421, Aug 2012.

Daniel Merkle and Martin Middendorf. Reconstruction of the cophylogenetic history of related phylogenetic trees with divergence timing information. *Theory Biosci*, 123:277–299, Apr 2005.

B. Mirkin, I. Muchnik, and T.F. Smith. A biologically consistent model for comparing molecular phylogenies. *J Comput Biol*, 2:493–507, 1995.

László G. Nagy, Robert Riley, Philip J. Bergmann, Krisztina Krizsán, Francis M. Martin, Igor V. Grigoriev, Dan Cullen, and David S. Hibbett. Genetic bases of fungal white rot wood decay predicted by phylogenomic analysis of correlated gene-phenotype evolution. *Mol Biol Evol*, 34:35–44, Jan 2017.

L. Nakhleh. Evolutionary phylogenetic networks: models and issues. In L. Heath and N. Ramakrishnan, editors, *The Problem Solving Handbook for Computational*, pages 125–158. Springer, 2010.

Luay Nakhleh. Computational approaches to species phylogeny inference and gene tree reconciliation. *Trends Ecol Evol*, 28:719–728, Dec 2013.

Luay Nakhleh and Derek Ruths. Gene trees, species trees, and species networks. In R. Guerra and D. Goldstein, editors, *Meta-analysis and Combining Information in Genetics and Genomics*, pages 275–293. Chapman & Hall/CRC Mathematical and Computational Biology, Boca Raton, FL, USA, 2009.

Luay Nakhleh, Derek Ruths, and Wang Li-San. RIATA-HGT: A fast and accurate heuristic for reconstructing horizontal gene transfer. In *Proc. 11th International Computing and Combinatorics Conference (COCOON)*, Springer, LNCS 3595, pages 84–93, Berlin Heidelberg, 2005.

Masatoshi Nei. *Molecular Evolutionary Genetics*. Columbia University Press, 1987.

Thi Hau Nguyen, Vincent Ranwez, Stéphanie Pointet, Anne-Muriel Arigon Chifolleau, Jean-Philippe Doyon, and Vincent Berry. Reconciliation and local gene tree rearrangement can be of mutual profit. *Algorithms for Molecular Biology*, 8(1):12, 2013.

Emmanuel Noutahi, Magali Semeria, Manuel Lafond, Jonathan Seguin, Bastien Boussau, Laurent Guéguen, Nadia El-Mabrouk, and Eric Tannier. Efficient gene tree correction guided by genome evolution. *PLoS One*, 11:e0159559, Aug 2016.

H. Ochman, J. G. Lawrence, and E. A. Groisman. Lateral gene transfer and the nature of bacterial innovation. *Nature*, 405(6784):299–304, May 2000.

S. Ohno. *Evolution by genome duplication.* Berlin: Springer Verlag, 1970.

Y. Ovadia, D. Fielder, C. Conow, and R. Libeskind-Hadas. The cophylogeny reconstruction problem is NP-complete. *J Comput Biol*, 18:59–65, Jan 2011.

R.D.M. Page. Maps between trees and cladistic analysis of historical associations among genes, organisms, and areas. *Syst Biol*, 43(1):58–77, Mar 1994.

P. Pamilo and M. Nei. Relationships between gene trees and species trees. *Mol Biol Evol*, 5 (5):568–583, 1988.

L. Patthy. Modular assembly of genes and the evolution of new functions. *Genetica*, 118 (2-3):217–231, Jul 2003.

Simon Penel, Anne-Muriel Arigon, Jean-François Dufayard, Anne-Sophie Sertier, Vincent Daubin, Laurent Duret, Manolo Gouy, and Guy Perrière. Databases of homologous gene families for comparative genomics. *BMC Bioinformatics*, 10 Suppl 6:S3, Jun 2009.

M. Desmond Ramirez, Autum N. Pairett, M. Sabrina Pankey, Jeanne M. Serb, Daniel I. Speiser, Andrew J. Swafford, and Todd H. Oakley. The last common ancestor of most bilaterian animals possessed at least nine opsins. *Genome Biol Evol*, 8:3640–3652, Dec 2016.

Matthew D. Rasmussen and Manolis Kellis. A Bayesian approach for fast and accurate gene tree reconstruction. *Mol Biol Evol*, 28:273–290, Jan 2011.

Aaron O. Richardson and Jeffrey D. Palmer. Horizontal gene transfer in plants. *J Exp Bot*, 58:1–9, Oct 2007.

F. Ronquist. Reconstructing the history of host parasite associations using generalised parsimony. *Cladistics*, 11(1):73–89, 1995.

Celine Scornavacca, Edwin Jacox, and Gergely J. Szöllõsi. Joint amalgamation of most parsimonious reconciled gene trees. *Bioinformatics*, 31:841–848, Mar 2015.

Joel Sjöstrand, Bengt Sennblad, Lars Arvestad, and Jens Lagergren. DLRS: gene tree evolution in light of a species tree. *Bioinformatics*, 28:2994–2995, Nov 2012.

Joel Sjőstrand, Ali Tofigh, Vincent Daubin, Lars Arvestad, Bengt Sennblad, and Jens Lagergren. A Bayesian method for analyzing lateral gene transfer. *Systematic Biology*, 63(3): 409, 2014.

J. G. Skellam. The frequency distribution of the difference between two Poisson variates belonging to different populations. *J R Stat Soc Ser A*, 109:296, 1946.

M Stolzer, H Lai, M Xu, D Sathaye, Vernot B, and D Durand. Inferring duplications, losses, transfers, and incomplete lineage sorting with non-binary species trees. *Bioinformatics*, 28:i409–i415, 2012a.

Maureen Stolzer. *Data Analysis Project: Inferring Rates of Domain Shuffling Using a Birth-Death and Gain Model*. PhD thesis, Carnegie Mellon University, Jan 2011.

Maureen Stolzer. *Phylogenetic Inference for Multidomain Proteins*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, Aug 2012.

Maureen Stolzer, Han Lai, Minli Xu, Deepa Sathaye, Benjamin Vernot, and Dannie Durand. Inferring duplications, losses, transfers, and incomplete lineage sorting with non-binary species trees. In *ECCB*, 2012b.

Maureen Stolzer, Katherine Siewert, Han Lai, Minli Xu, and Dannie Durand. Event inference in multidomain families with phylogenetic reconciliation. *BMC Bioinformatics*, 16(14):S8, 2015.

Krister M. Swenson, Andrea Doroftei, and Nadia El-Mabrouk. Gene tree correction for reconciliation and species tree inference. *Algorithms Mol Biol*, 7:31, Nov 2012.

Gergely J. Szöllosi, Bastien Boussau, Sophie S. Abby, Eric Tannier, and Vincent Daubin. Phylogenetic modeling of lateral gene transfer reconstructs the pattern and relative timing of speciations. *Proc Natl Acad Sci U S A*, 109:17513–17518, Oct 2012.

Gergely J. Szöllősi, Wojciech Rosikiewicz, Bastien Boussau, Eric Tannier, and Vincent Daubin. Efficient exploration of the space of reconciled gene trees. *Syst Biol*, 62:901–912, Nov 2013a.

Gergely J. Szöllősi, Wojciech Rosikiewicz, Bastien Boussau, Eric Tannier, and Vincent Daubin. Data from: Efficient exploration of the space of reconciled gene trees, 2013b. Dryad Digital Repository. http://dx.doi.org/10.5061/dryad.pv6df.

F. Tajima. Evolutionary relationship of DNA sequences in finite populations. *Genetics*, 105 (2):437–460, Oct 1983.

N. Takahata and M. Nei. Gene genealogy and variance of interpopulational nucleotide differences. *Genetics*, 110(2):325–344, Jun 1985.

Paul D. Thomas. GIGA: a simple, efficient algorithm for gene tree inference in the genomic age. *BMC Bioinformatics*, 11:312, Jun 2010.

J. W. Thornton and R. DeSalle. A new method to localize and test the significance of incongruence: detecting domain shuffling in the nuclear receptor superfamily. *Syst Biol*, 49(2):183–201, Jun 2000.

Ali Tofigh. *Using Trees to Capture Reticulate Evolution*. PhD thesis, Kunliga Tekniska Hogskolan School of Computer Science and Communication, Stockholm, Sweden, 2009.

Ali Tofigh, Michael Hallett, and Jens Lagergren. Simultaneous identification of duplications and lateral gene transfers. *TCBB*, 8:517–535, Mar/Apr 2011.

Hedvig Tordai, Alinda Nagy, Krisztina Farkas, Laszlo Banyai, and Laszlo Patthy. Modules, multidomain proteins and organismic complexity. *FEBS J*, 272(19):5064–5078, Oct 2005.

Todd J. Treangen and Eduardo P. C. Rocha. Horizontal transfer, not duplication, drives the expansion of protein families in prokaryotes. *PLoS Genet*, 7(1):e1001284, Jan 2011.

Ines Van Bocxlaer, Margo Maex, Dag Treer, Sunita Janssenswillen, Rik Janssens, Wim Vandebergh, Paul Proost, and Franky Bossuyt. Beyond sodefrin: evidence for a multi-component pheromone system in the model newt *Cynops pyrrhogaster* (Salamandridae). *Sci Rep*, 6:21880, Mar 2016.

Anke van Rijk and Hans Bloemendal. Molecular mechanisms of exon shuffling: illegitimate recombination. *Genetica*, 118:245–249, Jul 2003.

Deepti Varshney, Akanksha Jaiswar, Alok Adholeya, and Pushplata Prasad. Phylogenetic analyses reveal molecular signatures associated with functional divergence among subtilisin like serine proteases are linked to lifestyle transitions in Hypocreales. *BMC Evolutionary Biology*, 16(1):220, Oct 2016.

Alejandra Vasco, Tynisha L. Smalls, Sean W. Graham, Endymion D. Cooper, Gane Ka-Shu Wong, Dennis W. Stevenson, Robbin C. Moran, and Barbara A. Ambrose. Challenging the paradigms of leaf evolution: Class III HD-Zips in ferns and lycophytes. *New Phytol*, 212:745–758, Nov 2016.

Benjamin Vernot, Maureen Stolzer, Aiton Goldman, and Dannie Durand. Reconciliation with non-binary species trees. *J Comput Biol*, 15:981–1006, Oct 2008.

Albert J. Vilella, Jessica Severin, Abel Ureta-Vidal, Li Heng, Richard Durbin, and Ewan Birney. EnsemblCmpara GeneTrees: Complete, duplication-aware phylogenetic trees in vertebrates. *Genome Res*, 19:327–335, Feb 2009.

Ilan Wapinski, Avi Pfeffer, Nir Friedman, and Aviv Regev. Natural history and evolutionary principles of gene duplication in fungi. *Nature*, 449:54–61, Sep 2007.

André Wehe, Mukul S. Bansal, J. Gordon Burleigh, and Oliver Eulenstein. DupTree: a program for large-scale phylogenetic analyses using gene tree parsimony. *Bioinformatics*, 24:1540–1541, Jul 2008.

Jennifer H. Wisecaver, William G. Alexander, Sean B. King, Chris Todd Hittinger, and Antonis Rokas. Dynamic evolution of nitric oxide detoxifying flavohemoglobins, a family of single-protein metabolic modules in bacteria and eukaryotes. *Molecular Biology and Evolution*, 33(8):1979, 2016.

Guangchuang Yu, Li-Gen Wang, Yanyan Han, and Qing-Yu He. clusterProfiler: an R package for comparing biological themes among gene clusters. *OMICS*, 16:284–287, May 2012.

Liwen Zhang, Zhengfu Zhou, Qiannan Guo, Like Fokkens, Márton Miskei, István Pócsi, Wei Zhang, Ming Chen, Lei Wang, Yamin Sun, Bruno G. G. Donzelli, Donna M. Gibson, David R. Nelson, Jian-Guang Luo, Martijn Rep, Hang Liu, Shengnan Yang, Jing Wang, Stuart B. Krasnoff, Yuquan Xu, István Molnár, and Min Lin. Insights into adaptations to a near-obligate nematode endoparasitic lifestyle from the finished genome of *Drechmeria coniospora*. *Sci Rep*, 6:23122, Mar 2016.

Chunfang Zheng, P. Kerr Wall, Jim Leebens-Mack, Victor A. Albert, Claude dePamphilis, and David Sankoff. The effect of massive gene loss following whole genome duplication on the algorithmic reconstruction of the ancestral populus diploid. *Comput Syst Bioinformatics Conf*, 7:261–271, 2008.

Yu Zheng and Louxin Zhang. Reconciliation with non-binary gene trees revisited. In Roded Sharan, editor, *Research in Computational Molecular Biolog*, volume 8394 of *LNCS*, pages 418–432. Springer International Publishing, 2014a.

Yu Zheng and Louxin Zhang. Are the duplication cost and Robinson-Foulds distance equivalent? *J Comput Biol*, 21:578–590, Aug 2014b.