



HAL
open science

Inferring incomplete lineage sorting, duplications, transfers and losses with reconciliations

Yao-Ban Chan, Vincent Ranwez, Celine Scornavacca

► **To cite this version:**

Yao-Ban Chan, Vincent Ranwez, Celine Scornavacca. Inferring incomplete lineage sorting, duplications, transfers and losses with reconciliations. *Journal of Theoretical Biology*, 2017, 432, pp.1-13. 10.1016/j.jtbi.2017.08.008 . hal-02154862

HAL Id: hal-02154862

<https://hal.science/hal-02154862>

Submitted on 18 Dec 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Inferring incomplete lineage sorting, duplications, transfers and losses with reconciliations

Yao-ban Chan^{a,*}, Vincent Ranwez^b, Céline Scornavacca^{c,d}

^a*School of Mathematics and Statistics, The University of Melbourne, Melbourne, Australia*

^b*Montpellier SupAgro, UMR AGAP, F-34060 Montpellier, France*

^c*Institut des Sciences de l'Evolution (ISEM), Université de Montpellier, Montpellier, France*

^d*Institut de Biologie Computationnelle (IBC), Montpellier, France*

Abstract

Gene trees and species trees can be discordant due to several processes. Standard models of reconciliations consider macro-evolutionary events at the gene level: duplications, losses and transfers of genes. However, another common source of gene tree-species tree discordance is incomplete lineage sorting (ILS), whereby gene divergences corresponding to speciations occur “out of order”. However, ILS is seldom considered in reconciliation models. In this paper, we devise a unified formal \mathbb{I} DTL reconciliation model which includes all the abovementioned processes. We show how to properly cost ILS under this model, and then give a fixed-parameter tractable (FPT) algorithm which calculates the most parsimonious \mathbb{I} DTL reconciliation, with guaranteed time-consistency of transfer events. Provided that the number of branches in contiguous regions of the species tree in which ILS is allowed is bounded by a constant, this algorithm is linear in the number of genes and quadratic in the number of species. This provides a formal foundation to the inference of ILS in a reconciliation framework.

Keywords: reconciliation, gene duplication, gene transfer, incomplete lineage sorting, parsimony

*Corresponding author

Email address: yaoban@unimelb.edu.au (Yao-ban Chan)

12 1. Introduction

13 Macro-evolutionary events at the species level (i.e., speciation) impact the genomes of
14 the individuals belonging to the involved species. Hence, the evolutionary history of a group
15 of species strongly influences the evolutionary history of its genes. However, even though
16 species evolution strongly shapes each gene history, it does not fully determine it, and the
17 discrepancy between the two histories provides clues about gene-specific evolutionary events
18 such as gene duplication, gene transfer and gene loss.

19 Many methods have been proposed to *reconcile* the (inferred) evolutionary history of a
20 gene (depicted as a gene tree) with that of the corresponding species (depicted as a species
21 tree), using gene-specific events. In general, these methods fall into two paradigms: prob-
22 abilistic methods (e.g., [1, 20]), which find the most likely reconciliation under a statistical
23 model of evolution, and parsimony-based methods (e.g., [8, 4, 7]), which minimise the num-
24 ber (or total cost under a penalisation scheme) of the gene-specific events. In this paper, we
25 concentrate on the latter paradigm for reasons of efficiency and scalability.

26 Gene transfers are particularly difficult to take into account due to the time constraints
27 they induce [6]. Thus, reconciliation methods differ mainly by the way they handle transfer
28 events. Some simply ignore them, relying on the fact that transfers almost never occur in a
29 large part of the animal kingdom [29, 27]. Some search for optimal reconciliations without
30 considering the time constraints induced by transfers and, if needed, they either modify the
31 inferred solution to satisfy these constraints — with no guarantee of global optimality — or
32 they check for *time-consistency* of the transfers *a posteriori* and return an optimal solution
33 that is time-consistent, but only if any exists [16, 23]. Finally, some fully handle transfer
34 events and the associated time constraints in polynomial time, but require that the dates of
35 speciations are provided [4, 7, 25].

36 In addition to discrepancies caused by duplications, transfers and losses, an additional
37 source of discordance between gene and species trees arises from incomplete lineage sorting
38 [13]. In theory, incomplete lineage sorting is not a true “gene event” such as a duplication or
39 a transfer, since nothing “happens” to the gene during incomplete lineage sorting. Still, it is
40 a phenomenon that can lead to a gene tree differing from the species tree containing it. In
41 order to explain how ILS affects gene histories, we recall how a speciation acts on populations.

42 A speciation can be seen as the division of a population into (two) sub-populations that

43 will evolve separately and hence *fix* potentially different gene variants (alleles) so that those
44 alleles are somehow *sorted* from the originating population in the two sub-populations that
45 eventually become the two new species. For instance, in Figure 1, the ancestral population
46 giving rise to species *B* and *C*, prior to the speciation, contains blue and green alleles for
47 the considered locus; the speciation leads to two populations, one containing only blue alleles
48 (species *B*) and the other only green alleles (species *C*).

49 Such a “sorting” is not instantaneous, and if another speciation event occurs soon after
50 the first one, a locus may be *incompletely sorted* at the time of the second speciation. In such
51 a case, we can observe — in the two new species originating from the second speciation —
52 individuals that carry genes whose most recent common ancestor predates the first speciation
53 event. This results in the appearance of the two speciation events being “swapped” in the
54 gene tree, as shown in Figure 1.

55 The likelihood of an ILS is mainly related to the ancestral effective population size, which
56 can be hard to estimate, and the time elapsed between the two or more successive speciation
57 events, corresponding to the branch length of a dated species tree. However, in theory, given
58 any species tree, all possible gene tree topologies where each species has exactly one copy of
59 the gene can be explained by ILS alone.

60 The existence of ILS as a reason for discordance between gene and species trees has been
61 known for some time, and is often used in species tree inference from gene trees [12, 5]. In these
62 cases, the multispecies coalescent, arising from Kingman’s coalescent in population genetics
63 [9, 10], provides a statistical model under which the likelihood of ILS can be evaluated.

64 Inference of ILS via reconciliation is less common. In a seminal paper, Maddison [13] sug-
65 gested the parsimonious criterion of minimising deep coalescences (MDC) for reconciliation,
66 where the total number of “extra lineages” in all branches is minimised. An algorithm to
67 solve this problem was constructed by Than and Nakhleh [24], and extended for the presence
68 of hybridization in [28].

69 These papers did not consider macro-events such as duplications and losses, and indeed
70 very few papers attempt to combine both ILS and macro-events in a unified framework.
71 Combining these events is relevant from a biological perspective, as recent studies have shown
72 that ILS and gene introgression (although not specifically LGT) can both occur in the history
73 of a species [11, 14, 15]. More generally, with the increasing availability of data and efficiency

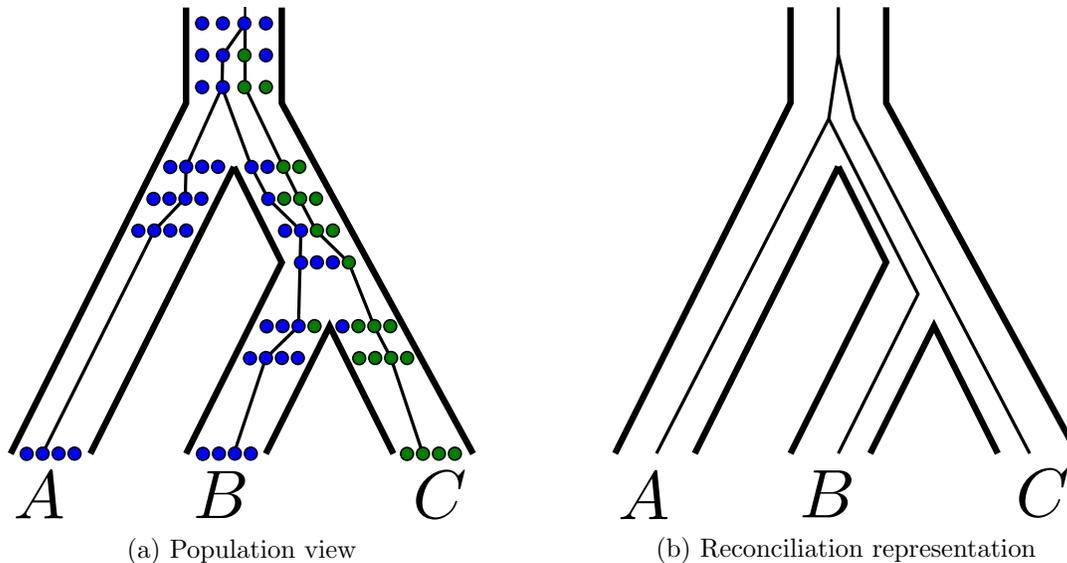


Figure 1: Impact of incomplete lineage sorting on simple populations of 4 haploid individuals. The originating population contains a single blue allele for the considered gene. First, a mutation leads to a new green allele at this locus, then a first speciation takes place, rapidly followed by a second one. As the blue and green alleles still co-exist when the second speciation takes place, both alleles still have a chance to be fixed in the resulting child species *B* and *C*. For these species, the history of this gene will hence differ from the species history due to ILS.

74 of algorithms, the number of species that can be considered in one tree is increasing rapidly.
 75 In consequence, even processes which mainly occur in different parts of the species tree must
 76 now be considered together, in order to capture all possible processes.

77 Of interest are a series of papers by Wu, Rasmussen and Kellis [21, 27], who model
 78 ILS together with duplications and losses using a coalescent model. They devised both a
 79 probabilistic algorithm (which was found to be very slow in practice), and a parsimony-based
 80 algorithm based on dynamic programming. However, their model does not include transfers
 81 and so does not need to consider the associated issues of time-consistency.

82 Another series of papers by Stolzer *et al.* [26, 23] formulated a full model with duplications,
 83 transfers, losses and ILS, and devised an algorithm to calculate the most parsimonious recon-
 84 ciliation for this model. Their algorithm starts by contracting short branches of the species
 85 tree into multifurcating nodes (polytomies). These are considered the only places where ILS
 86 can occur. Since ILS is not penalised in their model, discordance explainable by ILS is always
 87 associated to ILS. The remaining discrepancies are then explained by duplications, losses and
 88 transfers. However, their treatment of transfers does not guarantee a time-consistent rec-
 89 onciliation; this must be checked *a posteriori* and thus their algorithm may fail to return a

90 solution.

91 The precise complexity of the problem of finding an optimal reconciliation in the full model
92 with duplication, transfers, losses and ILS is unclear. Firstly, there can be slight variations
93 in the formulation of the model which may have an unknown and potentially drastic impact
94 on the complexity. Some information may be gleaned from complexity studies of various
95 special cases, which are necessarily no more complex than the full model. It is known that
96 the optimal \mathbb{D} TL reconciliation problem is NP-hard when the species tree is undated [18], but
97 is polynomial-time if the species tree is dated [7]. Likewise, it was proven recently [2] that
98 finding an optimal reconciliation in the duplication-loss-ILS model is also NP-hard, although
99 this complexity does not change if the species tree is dated or not. As the full model contains
100 this model as a special case (with transfers disallowed), it is likely that the problem we study
101 is also NP-hard, but as the two formulations do not correspond exactly, we cannot say this
102 with complete certainty.

103 In this paper, we formalise a model of reconciliation accounting for duplications, transfers,
104 losses and ILS. This \mathbb{ID} TL model is based on the standard \mathbb{D} TL model, formalised in [19],
105 with extensions to account for ILS. We then present an algorithm that calculates the most
106 parsimonious reconciliation for our model, and prove its correctness. This algorithm ensures
107 time-consistency through means of subdividing the species tree, as was done in [7], and thus
108 always returns an optimal time-consistent reconciliation. A detailed comparison with the
109 models and algorithms of [27] and [23] is also provided.

110 2. Preliminaries

111 Given a tree T , its node set, branches, and leaf set are respectively denoted $V(T)$, $E(T)$,
112 $L(T)$. The label of each leaf u is a name (typically an extant gene or species) or an identifier
113 associated with that leaf, denoted by $\mathcal{L}(u)$, while the set of labels of leaves of T is denoted
114 by $\mathcal{L}(T)$.

115 If T is rooted, we denote its root by $r(T)$. Given a node $u \in V(T)$, we denote its parent
116 by u_p , and the subtree of T rooted at u by T_u . Given two nodes u and v of T , we write $u \leq_T v$
117 ($u <_T v$) if and only if v is on the unique path from $r(T)$ to u (and $u \neq v$); in such a case, u is
118 said to be a (strict) descendant of v . The *height* of T , denoted $h(T)$, is the length, in nodes,
119 of the longest path from $r(T)$ to any leaf of T . From now on, unless otherwise specified, we

120 assume that all trees are rooted.

121 If a node in a tree T has more than two children, we call it a *polytomy*. If $u \in V(T)$ is not
122 polytomous, we denote its children by $\{u_l, u_r\}$; if u has just one child, u_r is understood to
123 be undefined. In this paper all trees are considered as unordered, so u_l and u_r are arbitrarily
124 assigned.

125 We define a *clade* of T as a set of leaves of T . The clade *generated* by the node u , denoted
126 $C(u)$, is the set $L(T_u)$. We define $\mathcal{C}(T)$ as the set of all clades generated by nodes in T ; for
127 a set \mathcal{T} of trees, $\mathcal{C}(\mathcal{T}) = \cup_{T \in \mathcal{T}} \mathcal{C}(T)$. The *LCA* of a clade is the internal node which is the
128 lowest common ancestor of the elements of the clade.

129 If u is a binary internal node, we define the *tripartition* generated by u , denoted by $\Pi(u)$, as
130 the clade triplet $(C(u), C(u_l), C(u_r))$. The latter two clades of a tripartition are a partition of
131 the first one — called *the parent clade* — since, for any internal node u , $C(u) = C(u_l) \cup C(u_r)$
132 and $C(u_l) \cap C(u_r) = \emptyset$. If u is an internal node with a single child, it generates the tripartition
133 $(C(u), C(u_l), \emptyset)$, while leaf nodes generate no tripartitions. We define $\Pi(T)$ as the set of all
134 tripartitions generated by nodes in T ; for a set \mathcal{T} of trees, $\Pi(\mathcal{T}) = \cup_{t \in \mathcal{T}} \Pi(T)$.

135 A tree T is said to be *dated* when there exists a *time function* $\theta_T : V(T) \rightarrow \mathbb{R}^+$ that
136 associates each of its nodes with a non-negative value so that, for any two nodes $x, y \in V(T)$,
137 if $y < x$ then $\theta_T(y) < \theta_T(x)$. Moreover, $\theta_S(x) = 0 \forall x \in L(T)$.

138 If a tree T is dated, then each clade it generates can be associated with the time of the
139 generating node. We therefore say that the tree generates *dated clades*, denoted by tuples
140 (C, t) . An internal node u generates the dated clade $(C(u), \theta_T(u))$, and *dated tripartitions*
141 are generated similarly. We denote by $\mathcal{C}_\theta(T)$ and $\Pi_\theta(T)$ respectively the sets of dated clades
142 and tripartitions generated by T .

143 We define the *subdivision* of a dated binary tree T with time function θ_T to be the unary-
144 binary tree T' obtained from T by adding a new unary node y on each branch $(x_p, x) \in E(T)$
145 such that there exists $z \in V(T)$ with $\theta_T(x) < \theta_T(z) < \theta_T(x_p)$; the time $\theta_{T'}(y)$ is set to
146 $\theta_T(z)$ (for all nodes u already in T , we have $\theta_{T'}(u) := \theta_T(u)$). These unary nodes are called
147 *artificial* nodes of T' . It is understood that the “+1” and “-1” operators, when applied to a
148 time t , indicate the lowest time in T' greater than t , and the highest time in T' lower than t ,
149 respectively.

150 We define a *gene tree* G as a tree where each leaf represents an extant gene. Similarly,

151 a *species tree* S is defined as a tree in which each leaf represents a distinct extant species.
152 Each extant gene is associated to its host species by a function $s : \mathcal{L}(G) \rightarrow \mathcal{L}(S)$, called the
153 *species labelling* of G . Note that s does not have to be either injective (several genes can be
154 contained in the same species due to duplication or transfers) or surjective (some species may
155 not contain any copy of the gene in question). The set of species labels of the leaves of G
156 is denoted $\mathcal{S}(G)$. In this paper, unless otherwise specified, we assume that gene and species
157 trees are rooted and binary. We will generally require that the species tree be dated, but the
158 gene tree need not be.

159 3. The model

160 In this section, we start by extending the DTL model formalised in [19] (an efficient
161 algorithm for this model was presented in [7] for a single rooted gene tree, and in [22] for
162 several, potentially unrooted, gene trees) to include incomplete lineage sorting of speciations,
163 to give an IDTL model. Then, we associate a cost to each ILS occurrence and present a
164 scoring scheme for IDTL reconciliations. The algorithm to compute a most parsimonious
165 reconciliation under this costing scheme will be given in the next section.

166 Firstly, we discuss (informally) how we model the events, to ease the understanding of the
167 formal definition (Definition 3). To construct a reconciliation, we map each gene tree node to
168 a sequence of dated clades of the species tree. We can consider each dated clade to represent
169 two things: a node in the subdivided species tree (essentially the “location” of the clade,
170 including its time), and a set of extant species into which the gene lineage will eventually be
171 “sorted” (descend), barring further events.

172 For example, consider Figure 2. Here, a simple I event (identical to that shown in Figure
173 1) causes the divergence between the genes in species B and C to occur before the speciation
174 at time 2. The reconciliation is as follows: the root gene is mapped to the dated clade
175 $(\{A, B, C\}, 2)$ and, after the initial I divergence, its descendants are mapped to $(\{A, B\}, 2)$
176 and $(C, 2)$, meaning that they will eventually be “sorted” into species A and B and species C
177 respectively, at the appropriate time. This is considered a valid mapping even though neither
178 of these two clades correspond to nodes of the species tree (or its subdivision). When these
179 lineages go forward in time, the clades to which they are mapped descend in the species tree,
180 and, eventually, will correspond to species tree nodes (here, at $(A, 1)$, $(B, 0)$ and $(C, 0)$). At

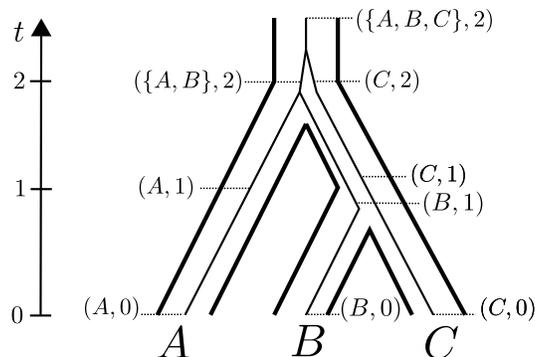


Figure 2: The thicker lines represent the species tree, while the thinner ones depict the gene tree. Dated clades represent the location and time of the gene lineage and also the set of species it will be sorted into. A , B , C are abbreviations for $\{A\}$, $\{B\}$, $\{C\}$ respectively.

181 this point, the ILS is considered as resolved.

182 We note that in the absence of ILS, the set of species that a gene lineage is “sorted” into
 183 is simply the descendants of the internal species node in which it resides. In this case, the
 184 only possible clades that a gene can be mapped to are the clades generated by the nodes of
 185 S' . These are clearly in bijection with the nodes of S' themselves. Therefore (for no ILS
 186 only) we can directly compare our model and previous models of reconciliations [7, 19] which
 187 map a gene to a sequence of nodes of S' . It is not too difficult to see that they are exactly
 188 equivalent in this case.

189 To model events, we consider the effect of each event on the two objects represented by
 190 a dated clade (C, t) , namely the leaf set (i.e., the set of descendant species) C and the time
 191 t . We first consider their impact on the leaf set C . In some situations, C may be partitioned
 192 into two subsets; we say this partitioning is *in accordance* with the species tree if it can be
 193 ascribed to a speciation. In other words, if C is partitioned into $C = C_1 \cup C_2$, then this
 194 partitioning is in accordance with the species tree if there exists a node $x \in V(S)$ such that
 195 $C \subseteq C(x)$, $C_1 \subseteq C(x_l)$, and $C_2 \subseteq C(x_r)$ (or vice versa with x_l and x_r interchanged). Then
 196 the events have the following impact on the leaf set (the examples in parentheses refer to the
 197 species tree in Figure 2, with the caveat that C represents the species named C , and not a
 198 clade):

- 199 • Speciations (\mathbb{S}) will create two gene lineages partitioning C into two subsets, in accor-
 200 dance with the species tree (e.g., $\{A, B, C\}$ can be partitioned into $\{A\}$ and $\{B, C\}$);
- 201 • Incomplete lineage sorting (\mathbb{I}) will create two gene lineages partitioning C into two

202 subsets, in a way that is not in accordance with the species tree (e.g., $\{A, B, C\}$ can be
203 partitioned into $\{A, B\}$ and $\{C\}$);

204 • Duplications (\mathbb{D}) will create two gene lineages, each having the same set of descendant
205 species as the original lineage;

206 • Transfers (\mathbb{T}) will create two gene lineages, one having the same set of descendant species
207 as the original clade, and the other corresponding to a node of the species tree at time
208 t (e.g., at $t = 1$, $\{B, C\}$ is a possible choice since $(\{B, C\}, 1)$ exists in the species tree);

209 • Speciation-losses (\mathbb{SL}) will result in a gene with a leaf set which is a subset of C , in
210 accordance with the species tree (e.g., $\{A, B, C\}$ can result in $\{B, C\}$ with $\{A\}$ lost);

211 • ILS-losses (\mathbb{IL}) will result in a gene with a leaf set which is a subset of C , in a way that
212 is not in accordance with the species tree (e.g., $\{A, B, C\}$ can result in $\{A, B\}$ with $\{C\}$
213 lost);

214 • Transfer-losses (\mathbb{TL}) will result in a gene corresponding to a node of the species tree at
215 time t ;

216 • Null events (\emptyset) will result in a gene having the same set of descendant species as the
217 original clade;

218 • \mathbb{C} events map an extant gene to an extant species containing the gene.

219 Note that we do not consider losses as separate events. Since it is impossible to distinguish
220 between a single loss and a subtree whose leaves are all lost, we only consider losses as part
221 of atomic \mathbb{SL} , \mathbb{IL} and \mathbb{TL} events.

222 The impact on the time of the events listed above is simple: the gene lineages created by
223 \mathbb{S} , \mathbb{SL} and \emptyset events have time $t - 1$, while in all other cases the new lineages have time t .

224 Before giving a formal definition of our model, we need to define the set of all possible
225 dated clades of a dated tree.

226 **Definition 1.** Consider a dated species tree (S, θ_S) . The expanded clade set of (S, θ_S) ,
227 denoted $\mathcal{C}'_\theta(S)$, is the set of all dated clades (C, t) such that $C \subseteq \mathcal{L}(S)$, $t \geq \theta_S(\text{LCA}(C))$, and
228 there exists some node $x \in V(S)$ with $\theta_S(x) = t$.

229 For example, for the dated species tree in Figure 2, we have $\mathcal{C}'_\theta(S) = (\{A\}, 0), (\{A\}, 1),$
 230 $(\{A\}, 2), (\{B\}, 0), (\{B\}, 1), (\{B\}, 2), (\{C\}, 0), (\{C\}, 1), (\{C\}, 2), (\{B, C\}, 1), (\{B, C\}, 2),$
 231 $(\{A, B\}, 2), (\{A, C\}, 2), (\{A, B, C\}, 2)$. Note that $\mathcal{C}'_\theta(S)$ equates to the set $\mathcal{C}_\theta(S')$ (the set of
 232 all dated clades generated by S') augmented with all clades that are possible due to ILS.

233 To aid interpretability, we now formalise the clade-to-node conversion:

234 **Definition 2.** We define $n : \mathcal{C}'_\theta(S) \rightarrow V(S')$ to be the function where $n(C, t) = x$ if:

- 235 • $x \geq LCA(C)$;
- 236 • $\theta_S(x) = t$.

237 It is easy to see that this mapping is well-defined: all nodes have only one ancestor that
 238 exists at a given time, so $n(\cdot)$ is unique, and, from the definition of $\mathcal{C}'_\theta(S)$, its value always
 239 exists. For example, for the species tree S depicted in Figure 2, we have that $n(\{A, B\}, 2)$ is
 240 equal to $r(S)$.

241 We are now ready to formally define a reconciliation, extending Definition 25 of [19]. As
 242 in previous models, a reconciliation can be thought of as “drawing a gene tree inside a species
 243 tree”; each branch of the gene tree forms a lineage which resides in the species tree, which
 244 must follow the species tree (i.e., descend into that species’ descendants), but may also be
 245 affected by gene-specific events (\mathbb{D} , \mathbb{T} , \mathbb{L}), and ILS.

246 **Definition 3** (Reconciliation). Consider a gene tree G and a dated species tree (S, θ_S) . Let
 247 $\alpha : V(G) \rightarrow \cup_{i=1}^\infty [\mathcal{C}'_\theta(S)]^i$ be a function which maps each node of G to an ordered sequence of
 248 dated clades in $\mathcal{C}'_\theta(S)$ of length at least 1. Let $\alpha_i(u)$ denote the i th element of $\alpha(u)$. Then α
 249 is a reconciliation between G and (S, θ_S) if and only if exactly one of the following mutually
 250 exclusive cases occurs for each element $\alpha_i(u)$ (with $(C, t) := \alpha_i(u)$ and $x := n(C, t)$ in the
 251 following):

- 252 • $\alpha_i(u)$ is the last element of $\alpha(u)$ and exactly one of the cases below is true:
 - 253 1. $x \in L(S')$, $u \in L(G)$, and $\mathcal{L}(x) = s(\mathcal{L}(u))$; (\mathbb{C} event)
 - 254 2. x is not artificial and $\{\alpha_1(u_l), \alpha_1(u_r)\} = \{(C \cap C(x_l), t - 1), (C \cap C(x_r), t - 1)\}$,
 255 with $C \cap C(x_l), C \cap C(x_r) \neq \emptyset$; (\mathbb{S} event)
 - 256 3. $\alpha_1(u_l) = \alpha_1(u_r) = (C, t)$; (\mathbb{D} event)

- 257 4. $\alpha_1(u_l) = (C, t)$ and $\alpha_1(u_r) = (C(y), t)$, where $y \in V(S')$ has $\theta_S(y) = t$ and $y \neq x$;
258 (T event)
- 259 5. $\{\alpha_1(u_l), \alpha_1(u_r)\} = \{(C_1, t), (C_2, t)\}$, where $C = C_1 \cup C_2$ and $C_1, C_2 \neq \emptyset$ and
260 $\nexists y \in V(S')$ with $\{C_1, C_2\} = \{C \cap C(y_l), C \cap C(y_r)\}$; (II event)
- 261 • otherwise, exactly one of the cases below is true:
- 262 1. x is not artificial and $\alpha_{i+1}(u) \in \{(C \cap C(x_l), t - 1), (C \cap C(x_r), t - 1)\}$, with
263 $C \cap C(x_l), C \cap C(x_r) \neq \emptyset$; (SL event)
- 264 2. $\alpha_{i+1}(u) = (C(y), t)$, where $y \in V(S')$ has $\theta_S(y) = t$ and $y \neq x$; (TL event)
- 265 3. $\alpha_{i+1}(u) = (C_1, t)$, where $C_1 \subset C$ and $\nexists y \in V(S')$ with $\{C_1, C \setminus C_1\} = \{C \cap$
266 $C(y_l), C \cap C(y_r)\}$; (III event)
- 267 4. $\alpha_{i+1}(u) = (C, t - 1)$, where one of $C \cap C(x_l), C \cap C(x_r)$ is \emptyset , and if $u = r(G)$, then
268 $i \neq 1$. (\emptyset event)

269 *Remarks:*.

- 270 1. We will sometimes write an event as “ $A \rightarrow B, C$ ”, where A, B and C are dated clades;
271 this means that for the affected gene v , we have $\alpha_\ell(v) = A$ (where $\ell := |\alpha(v)|$) and
272 $\{\alpha_1(v_l), \alpha_1(v_r)\} = \{B, C\}$.
- 273 2. When considering transfer targets, we only consider clades that are generated by S' and
274 not all the clades in the expanded clade set. The reason for this is that we do not allow
275 transfers into a species that fix in some descendants of the species but not in others.
276 In [27], this is referred to (albeit in the context of duplication) as *hemiplasy* and is also
277 not allowed. Thus, we assume that if a transfer occurs, it is fixed immediately in the
278 recipient (if subsequent children do not contain the gene, it must be due to a further loss
279 event). On the other hand, we do allow clades not generated by S' (i.e., incompletely
280 sorted alleles) to be sources of transfers.
- 281 3. We allow the root of the gene tree to be mapped initially to any node of the species
282 tree; we do not force it to be mapped to the root of the species tree.

283 In order to calculate a most parsimonious reconciliation, we must now define the cost of
284 a reconciliation. It is straightforward to cost events not involving ILS: we set δ, τ and λ to
285 be the cost of a duplication, a transfer and a loss respectively. Then S, D, T, SL and TL

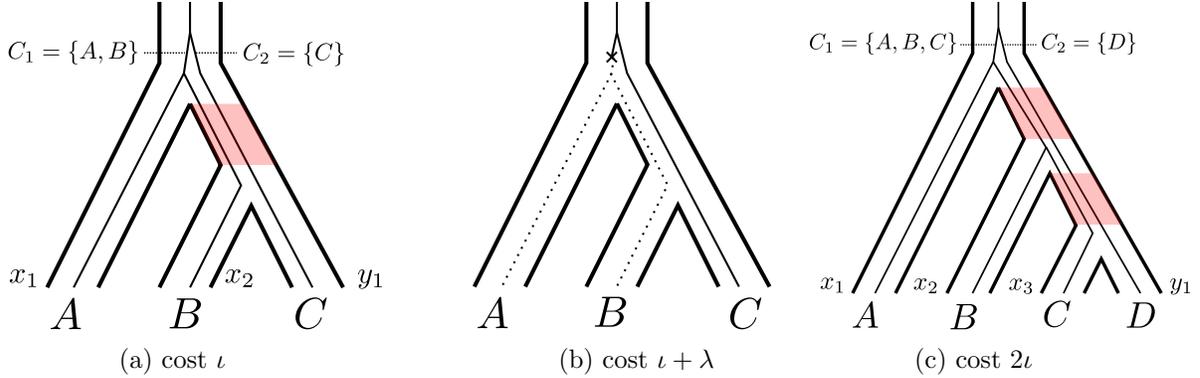


Figure 3: Costing ILS in various scenarios. In (a) the \mathbb{I} event implies that multiple alleles are present in one branch of the species tree, and therefore its cost is ι . In (b) one lineage is lost immediately after the initial divergence (cost λ), but the cost of the ILS event itself is the same as in (a), i.e. ι . In (c) the initial divergence implies that multiple alleles are present in two branches of the species tree and therefore the cost of this ILS event is 2ι . The notation of Definition 4 is shown in (a) and (c), and branches inferred to contain multiple alleles are shaded.

286 events cost 0, δ , τ , λ and $\tau + \lambda$ respectively. However, it is less straightforward to cost \mathbb{I}
 287 (and \mathbb{III}) events. This is because not all \mathbb{I} events are equally likely; an ILS which is resolved
 288 “quickly” is more likely to occur than one which induces gene tree-species tree discordance for
 289 a long period of time. We follow the MDC criterion of Maddison [13]; every \mathbb{I} event creates
 290 incompletely sorted (i.e., multiple) alleles in some branches of the species tree. We seek to
 291 minimise the number of these “extra lineages”, and therefore set the cost of an \mathbb{I} event to
 292 be proportional to the number of tree branches in which incompletely sorted alleles (deep
 293 coalescences) are created, barring further events. Note that further events may cause some
 294 alleles to be lost, but this does not affect the likelihood (and therefore the cost) of the \mathbb{I} event.

295 We define the cost of an \mathbb{I} event as follows:

296 **Definition 4** (Cost of an \mathbb{I} event). *Let $(C, t) \rightarrow (C_1, t), (C_2, t)$ be an \mathbb{I} event. Let $x_1, \dots, x_n \in$
 297 $V(S)$ be the set of nodes such that $C_1 = \cup_{i=1}^n C(x_i)$ and the $C(x_i)$ are maximal, i.e., there
 298 does not exist $x' \in V(S)$ with $C(x_i) \subset C(x') \subseteq C_1$. Define y_1, \dots, y_m similarly for C_2 . Then
 299 the cost of this \mathbb{I} event is the number of complete branches in S which are present in both a
 300 path from $n(C, t)$ to a $(x_i)_p$, and a path from $n(C, t)$ to a $(y_i)_p$, multiplied by ι .*

301 We define the cost in this way because the clade generated by each x_i (respectively y_i)
 302 is a subset of the set of species into which the gene is “sorted”, i.e., C_1 (respectively C_2).
 303 Thus the gene lineages descend from $n(C, t)$ to all $(x_i)_p$ and $(y_i)_p$. Branches which contain
 304 both lineages must contain multiple alleles and are costed accordingly. At the points $(x_i)_p$

305 and $(y_i)_p$, the genes are fixed in all descendant species; thus the species no longer contains
306 multiple alleles and we should not consider branches further down.

307 Note also that here we count branches in S , not S' ; that is, branches in the un-subdivided
308 species tree. This is logical as the presence of a speciation in a different part of the species
309 tree should not affect the cost of an ILS.

310 The cost of an III event is defined in a similar manner. See Figure 3 for some examples.

311 4. The algorithm

312 The model defined in the previous section allows ILS to occur in all parts of the tree.
313 Unfortunately, this produces an exponential explosion in the number of possible clades, let
314 alone reconciliations. It is impractical to find an optimal reconciliation under this model
315 without some restrictions.

316 In order to make our model tractable, we restrict ILS to only occur in certain branches.
317 More precisely, we only allow ILS to happen on branches of length not superior to a certain
318 threshold, which we denote by $ILSlength$. Furthermore, ILS can only happen in an internal
319 branch: leaf branches can never contain (observed) incompletely sorted alleles. We note that
320 this is not the only reasonable way to designate branches on which ILS may occur, and, in
321 theory, any method that designates certain branches that can contain ILS can be used in this
322 algorithm.

323 In this section we now describe an algorithm to compute the minimum cost of a reconcili-
324 ation between a gene tree G and a dated species tree (S, θ_S) , subject to the above restriction.
325 It is an extension of the algorithm of [7] with modifications to allow for incomplete lineage
326 sorting. In that algorithm, time-consistency of transfers is ensured by subdividing the species
327 tree and only allowing transfers within time “slices”. We also take this approach here.

328 The first step in the algorithm computes (under the length restriction):

- 329 • all possible dated clades, denoted $\mathcal{C}'_{\theta}(S)$;
- 330 • all possible tripartitions, denoted $\Pi'_{\theta}(S)$;
- 331 • the cost associated to these tripartitions, stored in the function $cost : \Pi'_{\theta}(S) \rightarrow \mathbb{R}^+$.

332 If ILS is not considered, these sets are simple to define: each clade of $\mathcal{C}'_{\theta}(S)$ corresponds to

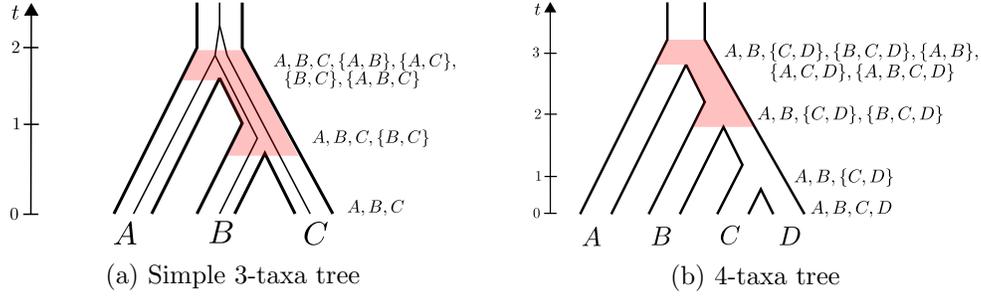


Figure 4: All possible clades for some simple trees. ILS branches are shaded. In (a), there is one ILS subtree with child clades A , B , C . In (b), there is one ILS subtree with child clades A , B , $\{C, D\}$. In both figures, clade dates are omitted for brevity and can be inferred from the positioning of the text.

333 a node of S' , and each tripartition of $\Pi'_\theta(S)$ corresponds to a node of S' , with elements that
 334 are (respectively) that node and its children.

335 However, when we take into account the possibility of ILS, the situation is much more
 336 complicated; we still start by subdividing the species tree and calculating the clades and
 337 tripartitions generated by it, but then we must augment these sets. To do so, we start by
 338 scanning the species tree and marking each internal branch that can contain ILS (i.e., has
 339 a length shorter than $ILSlength$). Each connected set of marked branches is considered as
 340 a single *ILS subtree* over which ILS can happen anywhere. Each child of the leaves of this
 341 subtree generates a clade, which we call *child clades*. Each child clade can be considered as a
 342 single unit with respect to this ILS subtree: it is impossible for ILS occurring in this subtree
 343 to split any of its child clades. On the other hand, the clade generated by the root of the ILS
 344 subtree can be resolved via ILS in any binary fashion that preserves the child clades. This
 345 means that any possible union of child clades is a possible clade of the ILS subtree, to which
 346 several times can be associated: the earliest possible time is the time of the root of the ILS
 347 subtree, while the latest is the time of the LCA of its elements.

348 We generate all possible dated clades for each ILS subtree as described above, and we
 349 add them to all clades generated by S' to form the set $\mathcal{C}'_\theta(S)$. See Figure 4 for some simple
 350 examples.

351 We next construct the set of possible tripartitions, which represent the possible ways in
 352 which a gene lineage can diverge due to speciation or ILS. Tripartitions corresponding to
 353 speciations not in ILS subtrees are defined as described in Section 2. We then consider each
 354 ILS subtree in turn, with each child clade of the subtree as an indivisible unit. Every clade

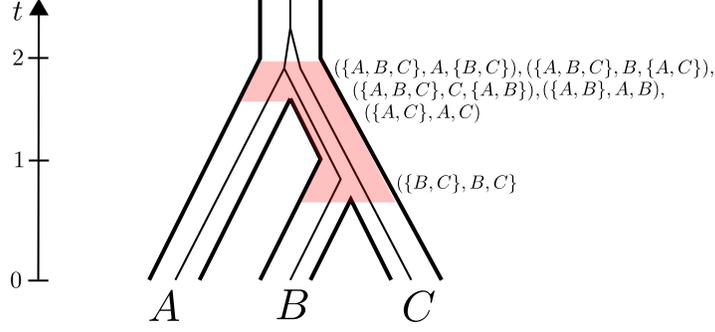


Figure 5: All possible tripartitions for the 3-taxon tree. Again, dates are omitted for brevity. Note that the tripartition $((\{B, C\}, 2), (B, 2), (C, 2))$ is not allowed as it corresponds to a speciation dated earlier than the LCA of B and C .

355 located in the ILS subtree can be split in any way which preserves the integrity of all child
 356 clades. If this partition can be ascribed to a speciation, then we assume it can only happen
 357 at the LCA of the species of its parent clade; otherwise, it can happen at any time that the
 358 parent clade exists in $\mathcal{C}'_\theta(S)$. In the latter case, this tripartition corresponds to an ILS. For a
 359 simple example, see Figure 5.

360 It still remains to describe how to calculate (and store) the costs of the ILS tripartitions.
 361 This can be done in a recursive manner for each ILS subtree. Consider an ILS tripartition
 362 $((C, t), (C_1, t), (C_2, t))$ (with $C = C_1 \cup C_2$). This tripartition will be located at $x := n(C, t)$,
 363 which we assume without loss of generality to be non-artificial. We must now consider which
 364 of the branches (x, x_l) and (x, x_r) contain alleles due to this tripartition (where x_l, x_r are the
 365 descendants of x in S). If both $C_1 \cap C(x_l)$ and $C_2 \cap C(x_l)$ are non-empty, then there will be
 366 alleles in (x, x_l) ; otherwise there will not, incurring no cost. If there are alleles, this adds a cost
 367 of ι to the tripartition. We then calculate the remaining number of branches with unsorted
 368 alleles below x_l by recursing on the tripartition $((C \cap C(x_l), \theta(x_l)), (C_1 \cap C(x_l), \theta(x_l)), (C_2 \cap$
 369 $C(x_l), \theta(x_l)))$, which may have a cost of 0. The cost for the alleles in the branch (x, x_r) and
 370 descendants is calculated in an identical manner, and these costs are summed to obtain the
 371 entire cost for the tripartition. By calculating these costs in order of increasing size of C
 372 and increasing time, we can calculate the costs of all ILS tripartitions. See Figure 6 for an
 373 example.

374 The formal pseudocode to generate the clade and tripartition sets and ILS costs is given
 375 in Algorithm 1.

376 Once we have computed the clade and tripartition sets, we proceed in a fashion that is

Algorithm 1 COMPUTE $C'_\theta(S)$, $\Pi'_\theta(S)$ and the *cost* function for the dated species tree (S, θ_S) , given a non-negative length $ILSlength$.

```

1: Mark each internal branch of  $S$  with length  $\leq ILSlength$ .
2:  $C'_\theta(S), \Pi'_\theta(S) \leftarrow \emptyset$ 

3: for each leaf node  $x \in V(S)$  do ▷ initialise with leaf clades
4:    $C'_\theta(S) \leftarrow C'_\theta(S) \cup \{(C(x), 0)\}$ 
5: end for

6: for each non-leaf node  $x \in V(S)$  in reverse time order do ▷ calculate child clades of ILS subtrees
7:    $ILSclades(x) \leftarrow \emptyset$ 
8:   for each  $x_c \in \{x_l, x_r\}$  do
9:     if  $(x, x_c)$  is marked then
10:       $ILSclades(x) \leftarrow ILSclades(x) \cup ILSclades(x_c)$ 
11:     else
12:       $ILSclades(x) \leftarrow ILSclades(x) \cup \{C(x_c)\}$ 
13:     end if
14:   end for
15: end for

16: for each internal node  $r \in V(S)$  whose parent branch is not marked do ▷ ordinary node or root of ILS subtree
17:   for each non-empty subset  $\{C_1, \dots, C_n\} \subseteq ILSclades(r)$  in order of increasing size ( $n$ ) do
18:      $C \leftarrow \cup_{j=1}^n C_j$ 
19:      $v \leftarrow LCA(C)$ 
20:      $t_C \leftarrow \theta(v)$ 
21:     for each time  $t = t_C + 1, \dots, \theta(r)$  do
22:        $\Pi'_\theta(S) \leftarrow \Pi'_\theta(S) \cup \{((C, t), (C, t - 1), \emptyset)\}$  ▷ pass through artificial node or be sorted by a speciation
23:        $cost((C, t), (C, t - 1), \emptyset) \leftarrow 0$  ▷ no cost
24:     end for
25:     for each non-empty subset  $\{C_{i_1}, \dots, C_{i_m}\} \subset \{C_1, \dots, C_n\}$  do
26:        $C_1 \leftarrow \cup_{j=1}^m C_{i_j}$ 
27:        $C_2 \leftarrow C \setminus C_1$ 
28:       if  $\theta(LCA(C_1)) < t_C$  and  $\theta(LCA(C_2)) < t_C$  then
29:          $\Pi'_\theta(S) \leftarrow \Pi'_\theta(S) \cup \{((C, t_C), (C_1, t_C - 1), (C_2, t_C - 1))\}$  ▷ divergence corresponding to speciation
30:          $cost((C, t_C), (C_1, t_C - 1), (C_2, t_C - 1)) \leftarrow 0$  ▷ no cost
31:       else
32:         for each time  $t = t_C, \dots, \theta(r)$  do
33:            $\Pi'_\theta(S) \leftarrow \Pi'_\theta(S) \cup \{((C, t), (C_1, t), (C_2, t))\}$  ▷ divergence corresponding to ILS
34:            $w \leftarrow$  oldest non-artificial node that is  $\leq n(C, t)$  ▷ calculate ILS cost
35:           for  $(w_c, c_c) \in \{(w_l, c_l), (w_r, c_r)\}$  do
36:             if  $C_1 \cap C(w_c), C_2 \cap C(w_c) \neq \emptyset$  then
37:                $c_c \leftarrow cost((C \cap C(w_c), \theta(w_c)), (C_1 \cap C(w_c), \theta(w_c)), (C_2 \cap C(w_c), \theta(w_c))) + \iota$ 
38:             else
39:                $c_c \leftarrow 0$ 
40:             end if
41:           end for
42:            $cost((C, t), (C_1, t), (C_2, t)) \leftarrow c_l + c_r$ 
43:         end for
44:       end if
45:     end for
46:   end for
47: end for

48: for each tripartition  $\pi \in \Pi'_\theta(S)$  do ▷ assemble clade set
49:    $C'_\theta(S) \leftarrow C'_\theta(S) \cup \pi[1]$ 
50: end for

51: return  $C'_\theta(S), \Pi'_\theta(S), cost$ 

```

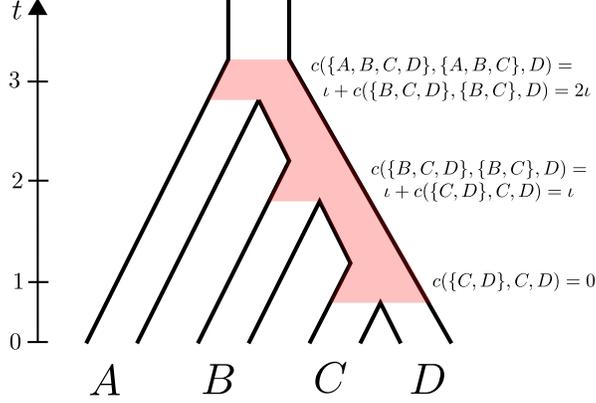


Figure 6: An example for costing the ILS tripartition $((\{A, B, C, D\}, 3), (\{A, B, C\}, 3), (D, 3))$ in a 4-taxa tree.

377 very similar to the original algorithm of [7], except that we must also apply the appropriate
 378 cost to ILS as specified in the previous section and calculated above. Of particular note is the
 379 calculation of the best transfer target for a given gene node v when initially mapped to a dated
 380 clade D ; this is stored in $BR(v, D)$ (where BR stands for Best Receiver). The pseudocode
 381 is given in Algorithm 2. We remind the reader here that the set of clades which are transfer
 382 targets is the set of clades generated by the subdivided species tree S' , i.e., $\mathcal{C}_\theta(S')$. We do not
 383 allow other clades (generated by ILS) to be transfer targets, in accordance with Definition 3.

384 We now show that our algorithm does indeed compute the optimal reconciliation cost
 385 (and, by backtracking, an optimal reconciliation).

386 **Theorem 5.** *Consider a gene tree G and dated species tree (S, θ_S) . Then $c(G, S)$ as computed
 387 by Algorithm 2 is the minimum cost of all reconciliations between G and (S, θ_S) .*

388 *Proof.* We begin by noting that in Algorithm 2, for a gene tree node v and species tree clade
 389 D , $c(v, D)$ calculates the minimum cost of reconciling the subtree of G generated by v to
 390 (S, θ_S) on the condition that $\alpha_1(v) = D$, i.e., v is first mapped to D .

391 For the most part, the correctness of the algorithm is then self-evident, as at each stage
 392 it merely enumerates all possible ways for the reconciliation to proceed (as according to
 393 Definition 3). Likewise, it is easy to see that the costs evaluated by Algorithm 1 are correct.

394 The only non-trivial issue arises from the treatment of TL and IL events. In the DTL
 395 model of [7], TL events had to be treated differently to prevent infinite loops, because TL
 396 events are the only events which do not change either the gene (node) or the time of the species
 397 (node). Thus when calculating $c(v, x)$, where v and x are nodes of G and S' respectively, the

Algorithm 2 COMPUTE $c(G, S)$ given positive costs δ , τ , and λ , respectively for \mathbb{D} , \mathbb{T} , \mathbb{L} events, a cost ι for ILS, and a non-negative length $ILSlength$.

```

1: Compute  $\mathcal{C}'_\theta(S)$ ,  $\Pi'_\theta(S)$  and  $cost$  according to Algorithm 1.

2: for each node  $v \in V(G)$  in bottom-up order do
3:   for  $t \in \{0, 1, \dots, h(S')\}$  in increasing order do

4:     for each dated clade  $D = (C, t) \in \mathcal{C}'_\theta(S)$ , in order of increasing size of  $C$  do
5:       for  $e \in \{\mathbb{S}, \mathbb{D}, \mathbb{T}, \emptyset, \mathbb{SL}, \mathbb{TL}\}$  do
6:          $c_e \leftarrow \infty$  ▷ initialise event costs
7:       end for

8:       if  $v \in L(G)$ ,  $D$  is a leaf clade and  $s(\mathcal{L}(v)) = \mathcal{L}(D)$  then ▷ see lines 3 – 5 of Algorithm 1
9:          $c(v, D) \leftarrow 0$  ▷  $\mathbb{C}$  event
10:        goto line 5
11:       end if

12:         $c_{\mathbb{D}} \leftarrow \min\{c_{\mathbb{D}}, c(v_l, D) + c(v_r, D) + \delta\}$  ▷  $\mathbb{D}$  event
13:         $c_{\mathbb{T}} \leftarrow \min\{c_{\mathbb{T}}, c(v_l, D) + c(v_r, BR(v_r, D)) + \tau, c(v_l, BR(v_l, D)) + c(v_r, D) + \tau\}$  ▷  $\mathbb{T}$  event

14:        for each dated tripartition  $\rho \in \Pi'_\theta(S)$  with  $\rho[1] = D$  do ▷ “divergence” in species
15:          if  $\rho[3] = \emptyset$  then
16:             $c_{\emptyset} \leftarrow \min\{c_{\emptyset}, c(v, \rho[2])\}$  ▷  $\emptyset$  event
17:          else
18:             $c_{\mathbb{SL}} \leftarrow \min\{c_{\mathbb{SL}}, c(v, \rho[2]) + \lambda + cost(\rho), c(v, \rho[3]) + \lambda + cost(\rho)\}$  ▷  $\mathbb{SL}$  or  $\mathbb{IL}$  event
19:          end if
20:        end for

21:        for each dated tripartition  $\rho \in \Pi'_\theta(S)$  with  $\rho[1] = D$  do
22:          if  $\rho[3] \neq \emptyset$  then
23:             $c_{\mathbb{S}} \leftarrow \min\{c_{\mathbb{S}}, c(v_l, \rho[2]) + c(v_r, \rho[3]) + cost(\rho), c(v_l, \rho[3]) + c(v_r, \rho[2]) + cost(\rho)\}$  ▷  $\mathbb{S}$  or  $\mathbb{I}$  event
24:          end if
25:        end for

26:         $c(v, D) \leftarrow \min\{c_e : e \in \{\mathbb{S}, \mathbb{D}, \mathbb{T}, \emptyset, \mathbb{SL}\}\}$  ▷ suboptimal cost: does not consider  $\mathbb{TL}$  events
27:      end for

28:      for each dated clade  $D = (C, t) \in \mathcal{C}'_\theta(S)$  with time  $t$  do
29:         $BR(v, D) \leftarrow \arg \min_{Y=(G,t) \in \mathcal{C}_\theta(S'), \text{ s.t. } G \not\supseteq C} c(v, Y)$  ▷ find the Best Receiver for transferring  $v$  at time  $t$ 
30:         $c_{\mathbb{TL}} \leftarrow c(v, BR(v, D)) + \tau + \lambda$  ▷  $\mathbb{TL}$  event
31:        if  $t \neq h(S')$  then
32:           $c_{2\mathbb{TL}} \leftarrow c(v, Z) + 2\tau + 2\lambda$ , where  $Z = (G, t) \in \mathcal{C}_\theta(S')$  s.t.  $G \supseteq C$  ▷  $\mathbb{TL}$ - $\mathbb{TL}$  to original species
33:        end if
34:         $c(v, D) \leftarrow \min\{c_{\mathbb{TL}}, c_{2\mathbb{TL}}, c(v, D)\}$  ▷ Final cost for  $c(v, D)$ 
35:      end for

36:    end for
37:  end for

38: return  $\min\{c(r(G), D) : D \in \mathcal{C}_\theta(S')\}$ 

```

398 cost of assigning a $\mathbb{T}\mathbb{L}$ event to v cannot be calculated together with the other events as there
399 is no guarantee that the subtree costs will already be calculated. In [7], this was accounted
400 for by observing that it is never most parsimonious to have two consecutive $\mathbb{T}\mathbb{L}$ events. Thus
401 the cost of assigning all events except $\mathbb{T}\mathbb{L}$ to v are found for all species tree nodes at the same
402 time as x , and then the cost of assigning a $\mathbb{T}\mathbb{L}$ event is found by calculating the best transfer
403 target for v from x based on these calculated costs; since two consecutive $\mathbb{T}\mathbb{L}$ events cannot
404 occur, this is optimal.

405 There is a similar but more complicated situation in our algorithm. Here both $\mathbb{T}\mathbb{L}$ and $\mathbb{I}\mathbb{L}$
406 events do not change either the gene (node) or time of the species (clade). However, an $\mathbb{I}\mathbb{L}$
407 event does reduce the size of the species clade, hence preventing the infinite loop problem and
408 ensuring the availability of needed cost values as long as smaller clades are processed first.

409 In addition, because it is possible for a gene node to be mapped to an incompletely sorted
410 species clade (i.e., one which is not generated by a species tree node), it is sometimes possible
411 for two consecutive $\mathbb{T}\mathbb{L}$ events to be most parsimonious. This can only happen if the original
412 species clade is incompletely sorted, then two consecutive $\mathbb{T}\mathbb{L}$ events transfer the gene to
413 another species (provided one exists) and then back to the original species (the gene is now
414 fixed in the entire species). This is the only scenario in which two consecutive $\mathbb{T}\mathbb{L}$ events can
415 be most parsimonious; note that three consecutive $\mathbb{T}\mathbb{L}$ events can never be most parsimonious.

416 In order to accommodate these, we apply a method that is similar to what is done in [7].
417 When calculating the cost of assigning a gene tree node v to a dated species tree clade D , we
418 calculate the cost of assigning any event except for $\mathbb{T}\mathbb{L}$ to v . Because we calculate these costs
419 in order of increasing size of D , we are sure that $\mathbb{I}\mathbb{L}$ events only reference costs which are
420 already calculated. After these costs are calculated for all clades of the same time, we then
421 calculate the costs of assigning a $\mathbb{T}\mathbb{L}$ event (line 29; the $G \not\subseteq C$ restriction prevents transfers
422 back to the same species), or two consecutive $\mathbb{T}\mathbb{L}$ events leading back to the original species
423 (line 32; here Z is the dated clade resulting from two $\mathbb{T}\mathbb{L}$ events back to the original species),
424 based on the previously calculated costs. These costs are then compared to the no- $\mathbb{T}\mathbb{L}$ cost
425 to calculate the final cost.

426 This provides an optimal cost because, in addition to it never being most parsimonious
427 to have additional $\mathbb{T}\mathbb{L}$ events, it is also never most parsimonious to have an $\mathbb{I}\mathbb{L}$ - $\mathbb{T}\mathbb{L}$ sequence
428 (removing the $\mathbb{I}\mathbb{L}$ will result in the same effect for a lower cost). Therefore we can calculate

429 the cost of assigning an \mathbb{I} event to v before calculating the cost of a \mathbb{T} event for v at the
430 same time.

431 We note that it is also never most parsimonious to have a sequence of consecutive \mathbb{I}
432 events, but this is not built into the algorithm. While this does not stop the algorithm from
433 being correct (we can never have an infinite loop of \mathbb{I} events only), it is a potential unused
434 source of optimisation.

435 □

436 The algorithm we have presented applies to the simplest case of a full, rooted, binary
437 gene tree reconciled to a dated (and thereby rooted), binary species tree. There are several
438 extensions to the algorithm of [7] in cases where these conditions are not met, and we discuss
439 the analogous extensions to our algorithm below.

440 *Amalgamating multiple gene trees*

441 In [22], an algorithm to amalgamate a set of rooted or unrooted gene trees \mathcal{G} while simul-
442 taneously reconciling with a dated species tree was presented. The basic idea is to cope with
443 gene tree uncertainty by considering not a single binary tree per gene but a set of realistic
444 alternative trees (e.g., those obtained by a bootstrap procedure). The amalgamation process
445 then selects compatible clades from this set of possible trees to build up a (possibly new) gene
446 tree minimising the reconciliation cost and made only of realistic clades.

447 We can adapt our algorithm to this case in a similar manner: instead of defining a rec-
448 onciliation as a mapping from the nodes of G to clades of S , we decompose the gene trees in
449 \mathcal{G} into their generated clades, then define a reconciliation as a mapping from gene clades to
450 species clades. The algorithm can then be used by matching each gene tripartition present
451 in the set of gene trees to either a species tree tripartition or a genetic event. This results
452 in a reconciliation which identifies the optimal gene tripartitions and thus defines an amalga-
453 mated gene tree which contains only tripartitions which are present in the set of gene trees.
454 Pseudocode for this extension is given in Appendix A.

455 *Unrooted gene trees*

456 The extension above can also be adapted for use with an unrooted gene tree (or amalga-
457 mating multiple unrooted gene trees). Here, we consider all clades and all gene tripartitions
458 present in all possible rootings of the gene tree(s), and then proceed as before.

459 *Undated species tree*

460 If the species tree is undated, we first assign each node a date of 0. Thus any branch can
461 contain a transfer to any other branch. In this case, we have no way of ensuring that the
462 optimal reconciliation produced is time-consistent; this must be checked *post hoc*, and if it is
463 found to contain a time paradox, another optimal reconciliation must be tried. Indeed, it is
464 possible that all optimal reconciliations are not time-consistent, in which case the algorithm
465 will fail.

466 Note that in this scenario, we must also have some alternative way of designating “ILS
467 branches”, as the species tree is undated. As observed before, it is impractical for all branches
468 to be ILS branches.

469 **5. Complexity**

470 Let k be the maximum polytomy degree (that is, the number of branches in the largest
471 ILS subtree plus 2) and n_k the number of polytomies.

472 We first count the number of clades. There are $O(|S|)$ nodes (counting internal nodes)
473 present in S , but due to subdivision these are replicated to $O(|S|^2)$ nodes in S' .

474 Now consider a single ILS subtree with k child clades. At the root there are $2^k - 1$ possible
475 clades to consider, but (some of) these clades are replicated throughout the ILS subtree. The
476 ILS subtree can have up to $k - 2$ non-artificial levels, but these may be subdivided from other
477 nodes outside the subtree; the best we can say is that the subtree has at most $|S|$ levels.
478 Therefore the number of clades generated by this tree is $O(|S|2^k)$, and the total number of
479 clades is

$$O(|S|^2 + n_k |S| 2^k).$$

480 In Algorithm 2, there are three nested loops: we loop over all nodes of G (of which there
481 are $O(|G|)$), then over all clades in $\mathcal{C}'_\theta(S)$ (the number of which we have calculated above),
482 then over all tripartitions corresponding to the species clade. (The complexity of the loop for
483 handling $\mathbb{T}\mathbb{L}$ events is clearly dominated by this loop over all tripartitions.) The maximum
484 number of these tripartitions for any clade is 2^k . Putting these three steps together, we find
485 that the total complexity for this algorithm is

$$O(|G|(|S|^2 + |S|n_k 2^k)2^k).$$

486 In comparison, the NOTUNG algorithm of Stolzer *et al.* [23] has an efficiency of

$$O(|G|(|S|+n_k2^k)^2(h_S+k)),$$

487 where h_S is the height of the species tree (which, in the worst case, is $O(|S|)$). This is slightly
488 worse than our algorithm, depending on the relative values of $|S|$ and k . In the case where k is
489 held fixed and $|S| \rightarrow \infty$, our algorithm is $O(|G||S|^2)$ and NOTUNG is $O(|G||S|^3)$. NOTUNG
490 additionally has other disadvantages as detailed in the next section.

491 If we are amalgamating m gene trees, the outermost loop would be over all clades appear-
492 ing in those trees, which requires at most $m|G|$ iterations instead of the $|G|$ previously for a
493 single gene tree, so the complexity of the algorithm is

$$O(m|G|(|S|^2+|S|n_k2^k)2^k).$$

494 If the gene tree(s) is unrooted, the complexity of the algorithm does not change; the
495 number of possible gene clades is multiplied by a constant factor of 2, and the number of
496 possible gene tripartitions by a factor of 3.

497 If the species tree is undated, we lose a factor of $|S|$ from the number of clades generated
498 by S' . We also replace the same factor from the ILS clades by the maximum height of an ILS
499 subtree (k). Therefore the complexity of the algorithm is

$$O(|G|(|S|+kn_k2^k)2^k),$$

500 with an extra factor of m if amalgamating m gene trees.

501 **6. Comparisons with other models**

502 Several algorithms to incorporate incomplete lineage sorting into reconciliation models
503 have been proposed before. In this section, we compare our model and algorithm with other
504 methods. It is important to note that often it is the model of incomplete lineage sorting
505 which differs slightly from author to author; each algorithm is formulated to solve the re-
506 spective model proposed, rather than a universally consistent model. This can make a direct
507 comparison between algorithms less meaningful.

508 Rasmussen and Kellis [21] proposed a model of incomplete lineage sorting (named DLCoal)
509 based on a coalescent model, and proposed a probabilistic reconciliation method (DLCoalRe-
510 con) which also incorporated duplication and loss. Wu, Rasmussen and Kellis [27] used the
511 same model, but devised a parsimonious method (DLCpar) instead, which is a more direct
512 relation to our method. A feature of their methods is that they keep direct track of the
513 (inferred) locus of the genes; this allows them to separate orthologous genes (arising from
514 speciation or ILS) from paralogous genes (arising from duplication).

515 Our model of ILS is largely similar the DLCoal model, albeit with some subtle differences.
516 While both models allow ILS to interact with other events (i.e., it is possible for genes which
517 are not fully resolved in a species to be duplicated or lost), the manner in which they interact
518 differs.

519 • The DLCoal model does not allow “hemiplasy”, i.e., if a gene duplicates, it is fixed
520 in all descendant species at the new locus. Thereafter, the duplicated gene evolves
521 independently from the original. If the original gene is fixed in the species at the time
522 of duplication (i.e., it is not part of an ILS), then this is identical to our model. On
523 the other hand, if the original gene is not yet fixed (due to ILS) in its species, in our
524 model the duplicated copy is enforced to remain in the same individuals as the original
525 gene; thus, barring further events, it will become fixed in exactly the same species as
526 the original gene. This is depicted in Figure 7. It is possible to enforce this in our
527 algorithm, because it keeps track of all the species into which each gene will eventually
528 be sorted. We also do not allow hemiplasy, but in the sense that the duplicated gene
529 cannot fix in some, but not all of the species that its parent fixes in.

530 An alternative way of viewing this is that our model does not allow recombination — if a
531 gene duplicates, the duplicated copies must evolve together rather than independently,
532 as they appear in the same individuals. In contrast, the DLCoal model allows free
533 recombination — once a gene appears at a different locus, it is considered for all intents
534 and purpose as a new gene. It is not immediately clear which model is more realistic,
535 or indeed if recombination should be allowed but penalised in some way. This would
536 introduce another layer of complexity to the model, and we do not consider it in this
537 paper.

538 These two different perspectives on duplication give slightly different costs for various

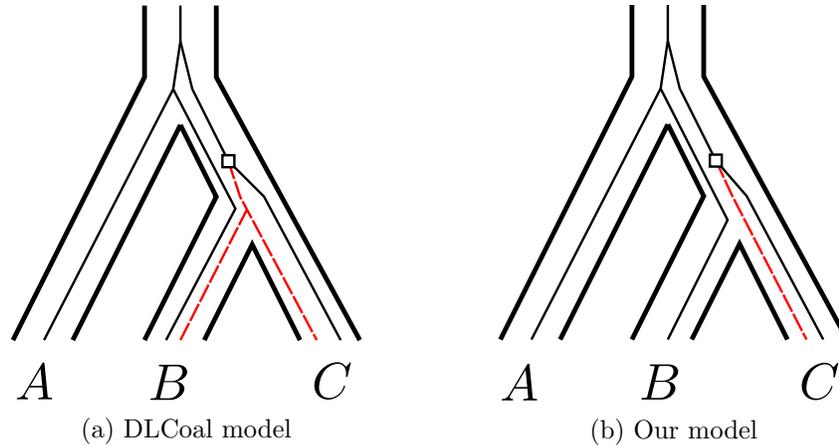


Figure 7: Different treatments of duplication within an ILS. In each case there is only one ILS and one duplication (denoted by a square). In (a), the duplicated gene (dashed) fixes in species *B* and *C*. In (b), the duplicated gene fixes only in species *C*, as its parent fixes only in *C*.

539 scenarios. For example, in Figure 7a, the DLCpar algorithm costs this scenario at $1\iota + 1\delta$,
 540 whereas we would not allow it all (we could mimic the effect of fixing the duplicated
 541 gene with two consecutive TL events, leading to a total cost of $1\iota + 1\delta + 2\tau + 2\lambda$,
 542 but this is a substantially different biological scenario). In contrast, in Figure 7b, our
 543 algorithm costs this scenario at $1\iota + 1\delta$, but DLCpar infers an extra loss, for a total cost
 544 of $1\iota + 1\delta + 1\lambda$.

- 545 • Because the DLCoal model is based on a coalescent perspective, it treats duplications
 546 from this viewpoint: running forwards in time, an allele is created (resulting in a di-
 547 vergence in the gene tree), which then simultaneously changes locus and becomes lost
 548 at the original locus. The end result is that the gene is duplicated at a new locus, but
 549 because there may be a delay between the creation of the allele and the change of lo-
 550 cus, it is possible to have incomplete lineage sorting between a duplication and a single
 551 speciation (see for example Figure 1C of [27]). In our model, we consider duplications
 552 to be instantaneous events which do not create alleles, and so this cannot happen.
- 553 • A similar scenario happens with the way losses interact with ILS. Because the DLCoal
 554 model arises from a coalescent perspective, it only “observes” (and thus costs) ILS when
 555 it infers two incompletely sorted alleles in the same locus (in one branch). It does not,
 556 and cannot, account for the possibility that a gene occurs in some of the population,
 557 while the rest of the population has no copies of the gene due to loss. This is allowed,

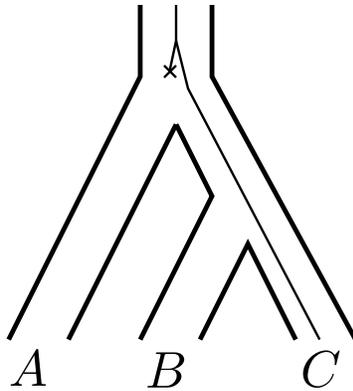


Figure 8: Loss within an ILS. Here is an ILS in the root species, but one allele (which would otherwise fix into species *A* and *B*) is immediately lost.

558 and accounted for, in our model. This is depicted in Figure 8; this scenario would be
 559 impossible in the DLCoal model.

- 560 • The DLCoal model allows incomplete lineage sorting to happen on all branches. This
 561 is necessary for this model, because as we discuss above, it does not require close speci-
 562 cations in order to have ILS. However, the probability of ILS between two speciations
 563 decreases quickly as the branch length increases. This is accounted for in DLCoalRecon,
 564 but not in DLCpar; there, the penalty for ILS is invariant to the length of the branch.
 565 Our algorithm allows ILS to occur only in the presence of rapid successive speciation
 566 events, i.e., over branches of small length (under a threshold).

567 While there are some differences in specifics, overall our model of ILS is largely similar
 568 to the DLCoal model. On the other hand, our reconciliation algorithm is entirely different
 569 to DLCpar, owing almost entirely to the fact that they do not consider transfers in their
 570 model. It is well known that including transfers in a reconciliation model makes it much
 571 harder; for example, the DL reconciliation model (without ILS) is easily solvable using the
 572 LCA mapping, but the DTL model is known to be NP-complete in general [25], and even
 573 with a dated species tree requires a polynomial-time algorithm [7]. DLCpar itself requires a
 574 dynamic programming algorithm to solve the ILS model with duplication and loss only, but
 575 the dynamic programming is iterated over the possible loci for genes in a branch, while each
 576 branch is solved more or less independently. Our algorithm iterates over the branches; while
 577 it is less efficient, this is an unavoidable result of the increased complexity from including
 578 transfers.

579 While our ILS model differs from DLCoal, it is more or less identical to that of Stolzer *et*
580 *al.* [23]. However, our method has some significant differences from theirs (NOTUNG):

- 581 • Our method constructs time-consistent solutions by incorporating the dates of the
582 species tree into the algorithm. In contrast, NOTUNG constructs reconciliations with-
583 out respect to time-consistency, then filters out time-inconsistent solutions *a posteriori*.
584 This can result in no solution being returned.
- 585 • Our method takes a binary species tree, and infers ILS (or other events) when there
586 is incongruence between the gene and species tree. NOTUNG first collapses all short
587 branches into polytomies, and allows ILS on only those branches. However the presence
588 of ILS is unpenalised and so there is no difference between when there is incongruence
589 with the species tree due to ILS and when there is no incongruence.
- 590 • Our method has a lower time complexity, as detailed in the previous section.

591 In addition, NOTUNG makes an implicit but unwarranted assumption that if a gene is present
592 in a species, then it must survive to at least one extant descendant of that species. This is
593 untrue, as it is possible that a gene can be transferred to another species and subsequently lost
594 (a TL event). This can cause NOTUNG to sometimes produce a suboptimal reconciliation.
595 See Figure 9: when we set the costs to $D = 2, T = 3, L = 1$, then NOTUNG infers the
596 reconciliation in Figure 9c for a cost of 7, whereas the reconciliation in Figure 9b has a lower
597 cost of 6. NOTUNG fails to infer this reconciliation because it contains a TL event. It is
598 known [3] that the DTL model simplifies significantly if TL events are disallowed.

599 The *mowgliNNI* algorithm of [17], and other algorithms for dealing with gene tree error,
600 could also be used in the context of incomplete lineage sorting, as it (heuristically) modifies
601 the gene trees using nearest-neighbour interchanges (NNI), which mimics the basic effect of
602 ILS. However, the two underlying problems are not equivalent, since *mowgliNNI* allows NNI
603 on pre-selected (unreliable) branches of the gene tree, whereas to correctly account for ILS,
604 NNI should be considered on pre-selected (short) branches of the species tree. Moreover, our
605 algorithm is an exact solution of the most parsimonious IDTL model, rather than a heuristic
606 to search a broader reconciliation space. Lastly, the effect of ILS is not limited solely to NNI,
607 especially in interactions with DTL events.

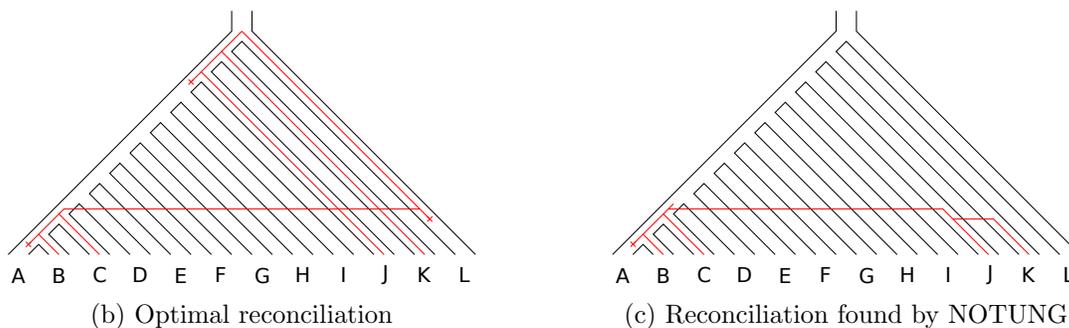
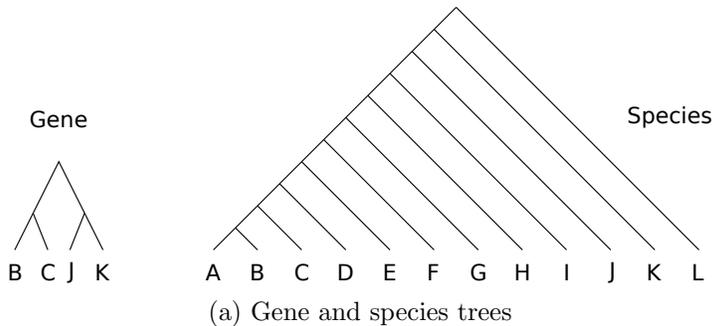


Figure 9: A scenario for which NOTUNG fails to identify the optimal reconciliation when costs are set to $\delta = 2, \tau = 3, \lambda = 1$.

608 **7. Conclusion**

609 In this paper, we have shown how to model incomplete lineage sorting in a reconciliation
 610 context. We have created a formal unified framework under which duplications, losses, trans-
 611 fers and ILS can all be accounted for. We have also developed an extension to the algorithm
 612 of Doyon *et al.* which calculates a globally most parsimonious IIDTL reconciliation, with guar-
 613 anteed time-consistency of transfers. This algorithm is efficient if ILS is not allowed on too
 614 many branches.

615 This work develops a foundation for the practical inference of ILS, by showing that it can
 616 be performed efficiently and how to do so. With these tools we can analyse real databases
 617 to measure how prevalent ILS is in evolutionary history and its relative importance to the
 618 other macro-events. We can also estimate the effect of ILS on the accuracy of phylogenetic
 619 and reconciliation inference. Finally, we can clearly distinguish between orthologous and
 620 paralogous genes. These analyses are the subject of future works.

621 The fixed-parameter tractable algorithm developed here is exponential only in k , the size
 622 of the largest ILS subtree. This means that it is a practical solution for most realistic cases

623 where ILS is mostly concentrated on few branches of the tree due to rapid successive speciation
624 events. Unlike the DLCpar algorithm, which allows ILS everywhere with no differentiation,
625 we allow it only on short branches. Ideally, we would like to allow ILS on all branches but
626 with a higher cost for ILS on longer branches; however, this would introduce another level of
627 complexity to the algorithm.

628 A potential way to make the algorithm more efficient is to only allow \mathbb{I} events which cost
629 less than a certain threshold, instead of or in addition to limiting branches on which ILS is
630 allowed. This would limit the number of possible clades generated by ILS and thus escape the
631 exponential dependence on maximum polytomy size, but again introduces more complexity
632 to the algorithm, and we have not explored it further here.

633 We lastly note that although ILS is not due to errors in gene tree inference, it is possible
634 that the algorithm here could be modified in order to find reconciliations which account for
635 gene tree error rather than ILS, as they both have similar effects on the gene tree (i.e., nearest
636 neighbour interchange).

637 **Acknowledgments**

638 We would like to thank Nicolas Galtier and Edwin Jacox for fruitful discussions. This work
639 was supported by the French Agence Nationale de la Recherche Investissements d’Avenir/Bioinformatique
640 (ANR-10-BINF-01-01, ANR-10-BINF-01-02, Ancestrome).

641 **References**

- 642 [1] L. Arvestad, A.-C. Berglund, J. Lagergren, and B. Sennblad. Gene tree reconstruction
643 and orthology analysis based on an integrated model for duplications and sequence evo-
644 lution. In *Proceedings of the Eighth Annual International Conference on Research in*
645 *Computational Molecular Biology*, pages 326–335. ACM, 2004.
- 646 [2] D. Bork, R. Cheng, J. Wang, J. Sung, and R. Libeskind-Hadas. On the computational
647 complexity of the maximum parsimony reconciliation problem in the duplication-loss-
648 coalescence model. *Algorithm. Mol. Biol.*, 12(1):6, 2017.
- 649 [3] Y. Chan, V. Ranwez, and C. Scornavacca. Exploring the space of gene/species reconcil-
650 iations with transfers. *J. Math. Biol.*, 71(5):1179–1209, 2015.

- 651 [4] L. A. David and E. J. Alm. Rapid evolutionary innovation during an Archaean genetic
652 expansion. *Nature*, 469(7328):93–96, 2010.
- 653 [5] J. H. Degnan and N. A. Rosenberg. Gene tree discordance, phylogenetic inference and
654 the multispecies coalescent. *Trends Ecol. Evol.*, 24(6):332–340, 2009.
- 655 [6] J. Doyon, V. Ranwez, V. Daubin, and V. Berry. Models, algorithms and programs for
656 phylogeny reconciliation. *Brief. Bioinform.*, 12(5):392–400, 2011.
- 657 [7] J.-P. Doyon, C. Scornavacca, K. Y. Gorbunov, G. J. Szöllősi, V. Ranwez, and V. Berry.
658 An efficient algorithm for gene/species trees parsimonious reconciliation with losses, du-
659 plications and transfers. In *Proceedings of the 2010 International Conference on Com-*
660 *parative Genomics*, RECOMB-CG'10, pages 93–108, Berlin, Heidelberg, 2011.
- 661 [8] P. Górecki and J. Tiuryn. DLS-trees: a model of evolutionary scenarios. *Theor. Comput.*
662 *Sci.*, 359(1):378–399, 2006.
- 663 [9] J. F. C. Kingman. The coalescent. *Stoc. Proc. Appl.*, 13(3):235–248, 1982.
- 664 [10] J. F. C. Kingman. On the genealogy of large populations. *J. Appl. Probab.*, pages 27–43,
665 1982.
- 666 [11] V. E. Kutschera, T. Bidon, F. Hailer, J. L. Rodi, S. R. Fain, and A. Janke. Bears in a
667 forest of gene trees: phylogenetic inference is complicated by incomplete lineage sorting
668 and gene flow. *Mol. Biol. Evol.*, 31(8):2004–2017, 2014.
- 669 [12] L. Liu, L. Yu, L. Kubatko, D. K. Pearl, and S. V. Edwards. Coalescent methods for
670 estimating phylogenetic trees. *Mol. Phylogenet. Evol.*, 53(1):320–328, 2009.
- 671 [13] W. P. Maddison. Gene trees in species trees. *Syst. Biol.*, 46(3):523–536, 1997.
- 672 [14] J. A. McGuire, C. W. Linkem, M. S. Koo, D. W. Hutchison, A. K. Lappin, D. I. Orange,
673 J. Lemos-Espinal, B. R. Riddle, and J. R. Jaeger. Mitochondrial introgression and
674 incomplete lineage sorting through space and time: phylogenetics of crotaphytid lizards.
675 *Evolution*, 61(12):2879–2897, 2007.

- 676 [15] B. S. McLean, D. J. Jackson, and J. A. Cook. Rapid divergence and gene flow at high
677 latitudes shape the history of holarctic ground squirrels (*urocitellus*). *Mol. Phylogenet.*
678 *Evol.*, 102:174–188, 2016.
- 679 [16] D. Merkle, M. Middendorf, and N. Wieseke. A parameter-adaptive dynamic program-
680 ming approach for inferring cophylogenies. *BMC Bioinformatics*, 11(1):1, 2010.
- 681 [17] T. H. Nguyen, V. Ranwez, S. Pointet, A.-M. A. Chifolleau, J.-P. Doyon, and V. Berry.
682 Reconciliation and local gene tree rearrangement can be of mutual profit. *Algorithm.*
683 *Mol. Biol.*, 8(1):12, 2013.
- 684 [18] Y. Ovadia, D. Fielder, C. Conow, and R. Libeskind-Hadas. The cophylogeny reconstruc-
685 tion problem is np-complete. *J. Comput. Biol.*, 18(1):59–65, 2011.
- 686 [19] V. Ranwez, C. Scornavacca, J.-P. Doyon, and V. Berry. Inferring gene duplications,
687 transfers and losses can be done in a discrete framework. *J. Math. Biol.*, pages 1–34,
688 2015.
- 689 [20] M. D. Rasmussen and M. Kellis. A Bayesian approach for fast and accurate gene tree
690 reconstruction. *Mol. Biol. Evol.*, 28(1):273–290, 2011.
- 691 [21] M. D. Rasmussen and M. Kellis. Unified modeling of gene duplication, loss, and coales-
692 cence using a locus tree. *Genome Res.*, 22(4):755–765, 2012.
- 693 [22] C. Scornavacca, E. Jacox, and G. J. Szöllősi. Joint amalgamation of most parsimonious
694 reconciled gene trees. *Bioinformatics*, 31(6):841–848, 2015.
- 695 [23] M. Stolzer, H. Lai, M. Xu, D. Sathaye, B. Vernet, and D. Durand. Inferring dupli-
696 cations, losses, transfers and incomplete lineage sorting with nonbinary species trees.
697 *Bioinformatics*, 28(18):i409–i415, 2012.
- 698 [24] C. Than and L. Nakhleh. Species tree inference by minimizing deep coalescences. *PLoS*
699 *Comput. Biol.*, 5(9):e1000501, 2009.
- 700 [25] A. Tofigh, M. Hallett, and J. Lagergren. Simultaneous identification of duplications and
701 lateral gene transfers. *IEEE-ACM T. Comput. Bi.*, 8(2):517–535, 2011.

- 702 [26] B. Vernot, M. Stolzer, A. Goldman, and D. Durand. Reconciliation with non-binary
703 species trees. *J. Comput. Biol.*, 15(8):981–1006, 2008.
- 704 [27] Y.-C. Wu, M. D. Rasmussen, M. S. Bansal, and M. Kellis. Most parsimonious recon-
705 ciliation in the presence of gene duplication, loss, and deep coalescence using labeled
706 coalescent trees. *Genome Res.*, 24(3):475–486, 2014.
- 707 [28] Y. Yu, R. M. Barnett, and L. Nakhleh. Parsimonious inference of hybridization in the
708 presence of incomplete lineage sorting. *Syst. Biol.*, page syt037, 2013.
- 709 [29] C. M. Zmasek and S. R. Eddy. A simple algorithm to infer gene duplication and speciation
710 events on a gene tree. *Bioinformatics*, 17(9):821–828, 2001.

711 **Appendix A. Pseudocode for multiple gene trees**

Algorithm 3 COMPUTE $c(\mathcal{G}, S)$ given positive costs δ , τ , and λ , respectively for \mathbb{D} , \mathbb{T} , \mathbb{L} events, a cost ι for ILS, and a non-negative length $ILSlength$.

```

1: Compute  $\mathcal{C}'_\theta(S)$ ,  $\Pi'_\theta(S)$  and  $cost$  according to Algorithm 1.
2:
3: for each clade  $C \in \mathcal{C}(\mathcal{G})$  in order of increasing size do
4:   for  $t \in \{0, 1, \dots, h(S')\}$  in increasing order do

5:     for each dated clade  $D = (G, t) \in \mathcal{C}'_\theta(S)$  with time  $t$ , in order of increasing size of  $D$  do
6:       for  $e \in \{\mathbb{S}, \mathbb{D}, \mathbb{T}, \emptyset, \mathbb{SL}, \mathbb{TL}\}$  do
7:          $c_e \leftarrow \infty$  ▷ initialise event costs
8:       end for

9:       if  $C$  and  $D$  are leaf clades and  $s(\mathcal{L}(C)) = \mathcal{L}(D)$  then ▷ see lines 3 – 5 of Algorithm 1
10:         $c(C, D) \leftarrow 0$  ▷  $\mathbb{C}$  event
11:        goto line 5
12:       end if

13:       for each tripartition  $\pi \in \Pi(C)$  do ▷ divergence in gene
14:         $c_{\mathbb{D}} \leftarrow \min\{c_{\mathbb{D}}, c(\pi[2], D) + c(\pi[3], D) + \delta\}$  ▷  $\mathbb{D}$  event
15:         $c_{\mathbb{T}} \leftarrow \min\{c_{\mathbb{T}}, c(\pi[2], D) + c(\pi[3], BR(\pi[3], D)) + \tau, c(\pi[2], BR(\pi[2], D)) + c(\pi[3], D) + \tau\}$  ▷  $\mathbb{T}$  event
16:       end for

17:       for each dated tripartition  $\rho \in \Pi'_\theta(S)$  with  $\rho[1] = D$  do ▷ “divergence” in species
18:        if  $\rho[3] = \emptyset$  then
19:           $c_{\emptyset} \leftarrow \min\{c_{\emptyset}, c(C, \rho[2])\}$  ▷  $\emptyset$  event
20:        else
21:           $c_{\mathbb{SL}} \leftarrow \min\{c_{\mathbb{SL}}, c(C, \rho[2]) + \lambda + cost(\rho), c(C, \rho[3]) + \lambda + cost(\rho)\}$  ▷  $\mathbb{SL}$  or  $\mathbb{IL}$  event
22:        end if
23:       end for

24:       for each tripartition  $\pi \in \Pi(C)$  do ▷ divergence in gene and species
25:        for each dated tripartition  $\rho \in \Pi'_\theta(S)$  with  $\rho[1] = D$  do
26:          if  $\rho[3] \neq \emptyset$  then
27:             $c_{\mathbb{S}} \leftarrow \min\{c_{\mathbb{S}}, c(\pi[2], \rho[2]) + c(\pi[3], \rho[3]) + cost(\rho), c(\pi[2], \rho[3]) + c(\pi[3], \rho[2]) + cost(\rho)\}$  ▷  $\mathbb{S}$  or  $\mathbb{I}$  event
28:          end if
29:        end for
30:       end for

31:        $c(C, D) \leftarrow \min\{c_e : e \in \{\mathbb{S}, \mathbb{D}, \mathbb{T}, \emptyset, \mathbb{SL}\}\}$  ▷ suboptimal cost: does not consider  $\mathbb{TL}$  events
32:     end for

33:     for each dated clade  $D = (G, t) \in \mathcal{C}'_\theta(S)$  with time  $t$  do
34:        $BR(C, D) \leftarrow \arg \min_{Y=(H,t) \in \mathcal{C}_\theta(S'), \text{ s.t. } H \not\subseteq G} c(C, Y)$  ▷ find the Best Receiver for transferring  $C$  at time  $t$ 
35:        $c_{\mathbb{TL}} \leftarrow c(C, BR(C, D)) + \tau + \lambda$  ▷  $\mathbb{TL}$  event
36:       if  $t \neq h(S')$  then
37:         $c_{2\mathbb{TL}} \leftarrow c(C, Z) + 2\tau + 2\lambda$ , where  $Z = (H, t) \in \mathcal{C}_\theta(S')$  s.t.  $H \supseteq G$  ▷  $\mathbb{TL}$ - $\mathbb{TL}$  to original species
38:       end if
39:        $c(C, D) \leftarrow \min\{c_{\mathbb{TL}}, c_{2\mathbb{TL}}, c(C, D)\}$  ▷ Final cost for  $c(C, D)$ 
40:     end for

41:   end for
42: end for

43: return  $\min\{c(L(\mathcal{G}), D) : D \in \mathcal{C}_\theta(S')\}$ 

```
