An Efficient Algorithm for the Reconciliation of a Gene Network and Species Tree

Yao-ban Chan
Melbourne Integrative Genomics / School of Mathematics and Statistics, The University of Melbourne, Australia

Abstract

The phylogenies of species and the genes they contain are similar but distinct, due to evolutionary events that affect genes but do not create new species. These events include gene duplication and loss, but also paralog exchange (non-allelic homologous recombination), where duplicate copies of a gene recombine. To account for paralog exchange, the evolutionary history of the genes must be represented in the form of a phylogenetic network. We reconstruct the interlinked evolution of the genes and species with reconciliations, which map the gene network into the species tree by explicitly accounting for these events. In previous work, we proposed the problem of reconciling a gene network and a species tree, but did not find an efficient solution for a general gene network. In this paper, we develop such a solution, and prove that it solves the most parsimonious reconciliation problem. Our algorithm is exponential only in the level of the gene network (with a base of 2), and we demonstrate that it is a practical solution through simulations. This allows, for the first time, a fine-grained study of the paralogy/orthology relationship between genes along their sequences.

2012 ACM Subject Classification Applied computing → Computational biology

Keywords and phrases Reconciliation, recombination, paralog exchange, phylogenetic network, gene duplication, gene loss

Digital Object Identifier 10.4230/LIPIcs.WABI.2024.3

Supplementary Material *Software*: `https://github.com/yao-ban/network-tree-reconciliation` archived at `swh:1:dir:7715f9c2c57631090f24e24e69a7fb3f99a5afc6`

1 Introduction

It is now well-established that the evolutionary histories of species and the genes they contain, although related, may differ. These discrepancies may be due to several causes, such as gene duplication and loss, lateral genetic transfer, and incomplete lineage sorting. Analysing the differences between the phylogenies of genes and species can give insights into the evolutionary forces shaping them both.

Two central problems in this area are those of reconciliation and species tree inference. In the former, we start with known phylogenies of both gene and species, and attempt to reconstruct a scenario which explains the differences between the two, using a number of specified gene evolution processes. In the latter, we start with several known gene phylogenies, and attempt to reconstruct a species tree by combining the information they contain. We will focus on the reconciliation problem in this paper. The two main paradigms of reconciliation inference are parsimony, where each evolutionary event is assigned a cost and the total cost minimised, and probabilistic, where the likelihood of the reconciliation under a statistical model of gene evolution is maximised. We focus on the parsimony paradigm here.

The most parsimonious reconciliation problem has a long history dating back to Goodman *et al.* [10]. Early models accounted only for gene duplication and loss, resulting in a model that was easy to solve with the lowest common ancestor (LCA) mapping [25], but not always realistic. These reconciliation models enabled the determination of whether genes within a family are paralogous or orthologous, with corresponding functional implications, by determining if their LCA is inferred to be a duplication or a speciation.

Subsequent models (such as [8]) also included lateral genetic transfer, increasing both their realism and the complexity of their solution. More recent models also account for further events and processes such as incomplete lineage sorting [20, 23, 4, 17], segmental duplications [7], gene conversion [13], transfer with replacement [12], and "failure to diverge" [9]. A review of recent developments in the field can be found in [19].

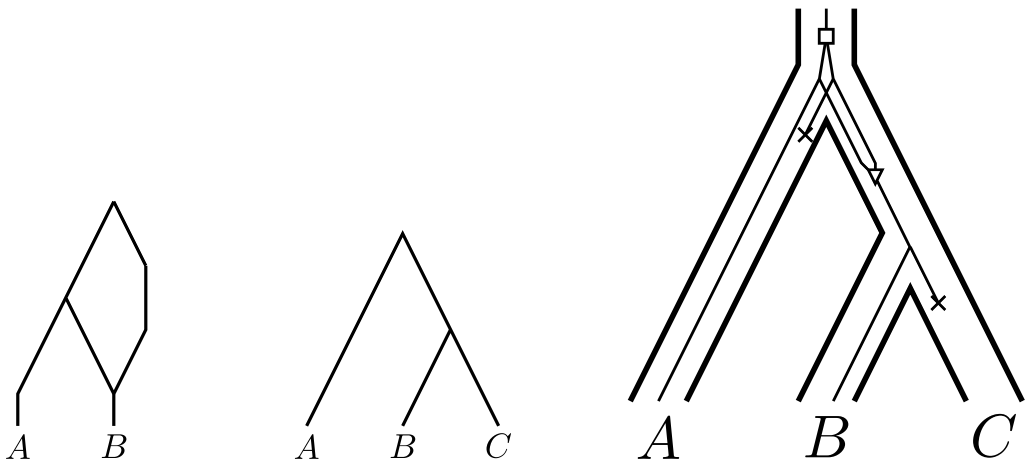
As models have increased in realism, there has been a growing awareness of the role that reticulate evolution has played in shaping gene and species histories. Such evolution cannot be represented in tree-like structures; instead, evolutionary histories must be represented by phylogenetic networks [15]. Some reconciliation algorithms have been devised for models where reticulations in the species phylogeny represent hybridisation [22] or lateral transfer [21] events. However, relatively little consideration (in phylogenetics) has been made of the fact that there is also reticulate evolution at the gene level.

Reticulate evolution in genes is well-studied in population genetics, where the evolutionary history of a recombining set of genes is expressed in the form of the Ancestral Recombination Graph (ARG; [11]). Inference of the ARG has long been considered a challenging problem, but recent developments (see [16, 2] for reviews) have made it possible to infer ARGs on a massive scale: some methods claim to be able to infer ARGs from genome-scale sequences for 10^5–10^6 individuals. These methods typically study allelic recombination within a single panmictic population, although some (e.g., ARGWeaver-D [14]) can also work with a known structured history of a small number of populations (typically a much smaller number than in a reconciliation study).

In [5], we suggested that another significant but underappreciated reticulate evolutionary process at the gene level is that of paralog exchange (or non-allelic homologous recombination). This process enables the diversification of gene duplicates, which can increase the fitness of the organism; for example, the duration of malaria infection is related to the diversity of the *var* genes of the malaria parasite *Plasmodium falciparum* [6]). We proposed the problem of reconciling a gene network to a species tree, taking into account gene duplication, loss, and paralog exchange (see Figure 1 for an example). We found that under certain conditions (that the gene network is tree-child), the problem could be solved efficiently by means of the so-called "LCA-HCA mapping", an extension of the traditional LCA mapping. However, we were not able to formulate an algorithm to efficiently solve the problem for a general gene network.

In this paper, a sequel to [5], we now provide an efficient algorithm to solve the problem of finding a most parsimonious reconciliation between a general gene network and a species tree. Although the theoretical difficulty of the problem is an open question, it shares some characteristics with known NP-hard problems in phylogenetics, in particular the dependence between gene lineages that prevents the use of standard dynamic programming methods. Our algorithm is exponential only in the level of the gene network, typically a much smaller number than the number of reticulations [1]. We perform simulations to show that this is a practical method for solving the network-tree reconciliation problem.

This also allows us, for the first time, to study the paralogy/orthology relationships between genes in a network. In particular, because a pair of genes may now have more than one LCA along their sequence, it is now possible that they may be paralogous in some parts of the sequence and orthologous in others, rather than necessarily being one or the other for the whole sequence. This opens the door to the fine-grained study of gene functions along their sequences.



(a) Gene network. **(b)** Species tree. **(c)** Reconciliation.

Figure 1 An example of a reconciliation between a gene network and a species tree. Gene nodes are mapped into the species tree to form the reconciliation. Each divergence node in the gene network can be mapped as either a duplication (square) or a speciation (divergence), while reticulations (triangle) must be between contemporary gene lineages in the same species. Losses (crosses) do not appear in the gene network and must immediately follow a speciation in the reconciliation.

2 Notation and preliminaries

We use the same notation as that of [5], reproduced here (with minor alterations) for completeness.

A phylogenetic network N on a label set $\mathcal{X}$ is a rooted, directed acyclic graph, containing nodes of four types:

- a root node has indegree 0 and outdegree 2;
- a divergence node has indegree 1 and outdegree 2;
- a reticulation node has indegree 2 and outdegree 1;
- a leaf node has indegree 1 and outdegree 0, and is labelled by an element of $\mathcal{X}$.

We consider root nodes to also be divergence nodes. We generally consider networks with a unique root node, although this is not necessary for the definition. This is commonly referred to as a *binary* phylogenetic network in the literature, but we drop the "binary" for the purposes of brevity. A phylogenetic tree is a phylogenetic network with no reticulation nodes.

For a network N, the sets of its nodes, branches, and leaves are denoted by $V(N)$, $E(N)$, and $L(N)$ respectively. The label of each leaf $u \in L(N)$ is denoted by $\mathcal{L}(u)$. The root node of N is denoted by $r(N)$.

For a non-root node $u \in V(N)$, we denote its parents by $\{u_p, u_q\}$, where u_q is undefined for nodes of indegree 1. The parents of a reticulation node are arbitrarily ordered. Likewise, the children of a non-leaf node $u \in V(N)$ are denoted by $\{u_l, u_r\}$, where u_r is undefined for nodes of outdegree 1, and the children are arbitrarily ordered otherwise. Given two nodes $u, v \in V(N)$, we write $u \leq_N v$ ($u <_N v$) if there exists a directed path from v to u in N (and $u \neq v$). We will drop the subscript N where it is clear. Two nodes $u, v \in V(N)$ are comparable if $u \leq v$ or vice versa.

The (unique) edge above a non-reticulation node $u \in V(N)$ is denoted by $e(u)$. We also include (for technical reasons) an artificial edge above the root node $r(N)$, with child $r(N)$ and no parent. Two edges $(u_1, v_1), (u_2, v_2) \in E(N)$ are comparable if they lie on the same directed path from the root, i.e., $u_2 \leq v_1$, or $v_2 \leq u_1$, or the edges are identical.

For a tree T, for a node $v \in V(T)$, T_v is the subtree of T consisting of v and all its descendants. (A similar concept can be defined for networks.) Given a set of leaf nodes $C \subseteq L(T)$, the lowest common ancestor (LCA) of C, denoted $LCA(C)$, is the unique minimal node $u \in V(T)$ such that $u \geq v$ for all $v \in C$. For two nodes $u, v \in V(T)$, the distance $d(u, v)$ is the number of edges on the unique path from u to v (although u and v need not be comparable, we only use d for comparable nodes).

A set of edges $E \subseteq E(N)$ is a cut set of N if its removal separates N into more than one connected component. For a cut set E, N_E is the network forest (set of networks) resulting from replacing all parent nodes of edges in E by separate root nodes, and taking the networks consisting only of these nodes and all their descendants (this may result in networks that have multiple root nodes).

A network is biconnected if removing any node does not disconnect the network. We do not consider the graph with two nodes and a single edge to be biconnected. A biconnected component of a network is a maximal subnetwork which is biconnected. It is well-known [15] that any network can be expressed as a tree of its biconnected components. A biconnected component B must have a unique root node, denoted $r(B)$. The set of edges from nodes in B to nodes not in B is denoted $out(B)$. A level-k network has at most k reticulation nodes in each biconnected component. A tree-child network [3] is a network in which every internal node has at least one child which is not a reticulation node.

A gene network (tree) is a phylogenetic network (tree) where each leaf node represents an extant gene in a gene family. Similarly, a species network (tree) is a phylogenetic network (tree) where each leaf node represents an extant species. Given a gene network/tree G and species network/tree S, each extant gene is associated to the species that contains it by a function $s : L(G) \to L(S)$, called the species labelling. For clarity, we will conventionally refer to the nodes of a gene network/tree as nodes and the nodes of a species network/tree as vertices.

Given a gene network G, species tree S, species labelling s, and a gene node $v \in V(G)$, we define $LCA(v)$ to be $LCA(s(L(G_v)))$; that is, the lowest common ancestor of all species containing descendants of the gene v.

A reconciliation and its cost are formally defined as follows.

▶ **Definition 1** (Reconciliation, [5]). *Let G be a gene network and S a corresponding species tree, with s the species labelling between G and S. A* reconciliation *between G and S is a function tuple (f, ϵ), where $f : V(G) \to V(S)$ and $\epsilon : V(G) \to \{\mathbb{S}, \mathbb{D}, \mathbb{R}, \mathbb{C}\}$ satisfy the following conditions for all $v \in V(G)$:*

- *If v is a divergence node, then either:*
  - *$\epsilon(v) = \mathbb{S}$ and $f(v_l) \leq f(v)_l$ and $f(v_r) \leq f(v)_r$, or vice versa (speciation); or*
  - *$\epsilon(v) = \mathbb{D}$ and $f(v_l), f(v_r) \leq f(v)$ (duplication);*
- *If v is a reticulation node, then $\epsilon(v) = \mathbb{R}$ and $f(v_l) \leq f(v)$;*
- *If v is a leaf node, then $\epsilon(v) = \mathbb{C}$ and $f(v) = s(v)$.*

▶ **Definition 2** (Cost of a reconciliation, [5]). *Let G be a gene network and S a corresponding species tree. Given a reconciliation (f, ϵ) between G and S, and costs $\delta, \lambda \geq 0$ for duplications and losses respectively, the* cost *of the reconciliation is $c(f, \epsilon) = n \cdot \delta + m \cdot \lambda$, where n is the number of duplications and m is the number of losses in the reconciliation. These are calculated by:*

- *$n = \big|\{v \in V(G) \,|\, \epsilon(v) = \mathbb{D}\}\big|$;*
- *$m = \sum_{(u,v) \in E(G)} \big[d(f(u), f(v)) - I(\epsilon(u) = \mathbb{S})\big]$, where $I(x) = 1$ if x is true and 0 otherwise.*

We note that reticulations have no cost as their number is fixed by the gene network.

The LCA-HCA ("lowest common ancestor – highest common ancestor") mapping is a specific reconciliation, defined as follows.

▶ **Definition 3** (LCA-HCA mapping, [5]). *Let G be a gene network and S a corresponding species tree, with s the species labelling between G and S. Define a reconciliation (f, ϵ) in the following way:*

1. *For all leaf nodes $v \in V(G)$, set $f(v) = s(v)$ and $\epsilon(v) = \mathbb{C}$.*
2. *For all divergence nodes $v \in V(G)$, set $f(v) = LCA(v)$. If $LCA(v) > LCA(v_l), LCA(v_r)$, then set $\epsilon(v) = \mathbb{S}$. Otherwise, set $\epsilon(v) = \mathbb{D}$.*
3. *Order the reticulation nodes of G in descending order, so that each node comes after all of its ancestors. For all reticulation nodes $v \in V(G)$ in this order, set $f(v)$ to be the maximal vertex such that:*
   - *$f(v) \geq LCA(v)$;*
   - *for $u \in \{v_p, v_q\}$, if $\epsilon(u) = \mathbb{S}$ then $f(v) < f(u)$; otherwise, $f(v) \leq f(u)$.*

   Set $\epsilon(v) = \mathbb{R}$.

We call the reconciliation defined this way the LCA-HCA mapping *between G and S.*

In this paper, we study the problem of finding a most parsimonious reconciliation between a gene network and species tree.

▶ **Problem 4** (Most Parsimonious Reconciliation, MPR, [5]). *Given a gene network G and a corresponding species tree S, and costs $\delta, \lambda \geq 0$ for duplications and losses respectively, find a reconciliation between G and S with the lowest cost.*

For completeness, we reproduce (in our notation) two results from [5] that will be used below.

▶ **Theorem 5** (Theorem 1, [5]). *Let G be a tree-child gene network and S a corresponding species tree. Then the LCA-HCA mapping from G to S is a most parsimonious reconciliation between G and S.*

▶ **Theorem 6** (Theorem 8, [5]). *Let G be a gene network and S a corresponding species tree. Then there exists a most parsimonious reconciliation (f, ϵ) between G and S such that, for each $v \in V(G)$ which is the root of a biconnected component:*

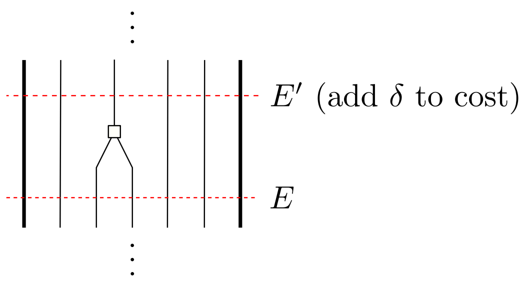
- *$f(v) = LCA(v)$;*
- *$\epsilon(v) = \mathbb{D}$ if the biconnected component has size greater than 1.*

3 The algorithm

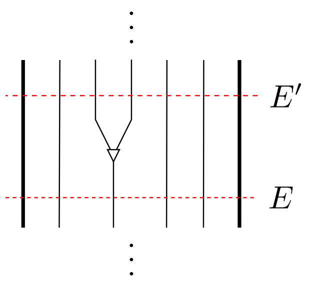
3.1 Description

We now formulate an algorithm to calculate the MPR between a gene network G and species tree S. This algorithm essentially uses the idea of "ancestral configurations" [24] in that it performs dynamic programming on the branches of the species tree, in contrast to standard reconciliation algorithms (e.g., for the DTL problem) which perform dynamic programming on the branches of the gene tree. As the gene phylogeny is a network, the latter approach no longer works.

For a set of gene branches $E \subseteq E(G)$, we define $c(E, x)$ to be the current minimum cost of a reconciliation of the network forest G_E to the species subtree S_x, where each edge in E is "mapped" to the edge above x; that is, for $(u, v) \in E$, $f(u) = x_p$ and $f(v) \leq x$. Intuitively,



(a) Contracting a divergence.



(b) Expanding a reticulation.

Figure 2 Possible operations for an edge set E mapped to a species x, resulting in a new edge set E'.

we can think of this as the cost of a partial reconciliation, where only gene nodes below E have assigned mappings into S, and must be mapped at x or below. We calculate this quantity recursively.

Firstly, from Theorem 6 we know that the root of each biconnected component of G must always be mapped to its LCA, so we can calculate the reconciliation (and thus the optimal cost) of each biconnected component separately. We thus calculate the cost $c(\{e(r(B))\}, LCA(r(B)))$ for each biconnected component B in bottom-up order. The initial condition for leaves (which are always biconnected components of size 1) is trivial, as they must be mapped to the corresponding species leaf with cost 0 (i.e., if $v \in L(G)$, then $c(\{e(v)\}, s(v)) = 0$).

Now consider a biconnected component B; from Theorem 5, we know that if B is tree-child, then the LCA-HCA mapping is an optimal reconciliation for the nodes of B, and thus the cost of its mapping can be easily found. Otherwise, we can assume that for each out-edge $e = (u, v) \in out(B)$, the cost $c(\{e\}, LCA(v))$ has been found by recursion. We now proceed by dynamic programming on the vertices of the species tree. At each species tree vertex $x \in V(S)$, if we have calculated a cost $c(E, x)$, we recursively calculate further costs at x in two ways:

- Contract a divergence (Figure 2a): if E contains the two child edges of a divergence node, replace them by the parent edge of that node and add a cost of δ, resulting in a new edge set E'. This corresponds to placing that node as a duplication on the species branch above x.
- Expand a reticulation (Figure 2b): if E contains the child edge of a reticulation node, replace it by the two parent edges of that node, resulting in a new edge set E'. This corresponds to placing that node on the species branch above x.

In either case, we update the minimum cost of (E', x) if the new cost is lower than the current minimum.

When we contract a divergence, we no longer need to consider the set E for the vertex x. Intuitively, this is because all divergences "sink" (are placed as low as possible, subject to the constraints of other nodes), so a divergence which can occur at a vertex will occur there. To keep track of this, we maintain an "active set" (denoted A) of (E, x) configurations that we must consider further. Contracting a divergence removes the original configuration (E, x) from the active set, replacing it with the new configuration (E', x). To maximise efficiency, we contract all possible divergences before we expand reticulations.

On the other hand, if we expand a reticulation we must still consider the original configuration (E, x), so we add the new configuration to the active set A but do not remove the original. Instead, we "mark" the reticulation in the original configuration (so that we

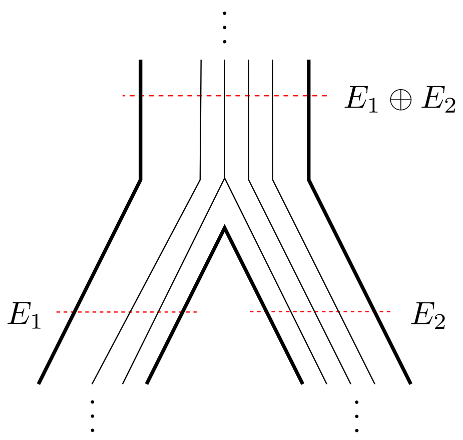


Figure 3 Joining two edge sets E_1 and E_2 mapped to the children of a species x, resulting in a new edge set $E_1 \oplus E_2$ mapped to x. (For neatness, losses are not shown.)

do not repeatedly expand the same reticulation). To improve efficiency, we can calculate a set of possible mappings $M_v \subseteq V(S)$ in an MPR for each reticulation node $v \in V(G)$. This may come, for example, from the MPNSR algorithm of [5], or simply by restricting v to be placed no higher than the root of its biconnected component. If we expand a reticulation and the reticulation cannot be mapped higher up in the species tree in any MPR, we can then remove (E, x) from the active set.

Once we have calculated as many costs $c(E, x)$ as possible at a given vertex, we proceed up the species tree. For a given species vertex x, we can calculate initial costs from the previously calculated costs at x_l and x_r. Given the costs $c(E_1, x_l)$ and $c(E_2, x_r)$, we merge the two sets E_1 and E_2 together by first taking their union and then contracting all divergences that arise from the union; we call the resulting set $E_1 \oplus E_2$ (Figure 3; formal definition below). These divergences are assigned as speciations at x, and as before must occur for the reconciliation to be most parsimonious. To yield the cost $c(E_1 \oplus E_2, x)$, we must add one loss for each edge in $E_1 \oplus E_2$ that is in either E_1 or E_2 (and thus must proceed with a speciation-loss event to a child of x). We must additionally add any out-edges of B which are known to be mapped to x.

▶ **Definition 7.** *Let $E_1, E_2 \subseteq E(G)$. Let*

$$V = \{v \in V(G) \,|\, (v, v_l) \in E_1 \text{ and } (v, v_r) \in E_2 \text{ or vice versa}\}.$$

Then we define

$$E_1 \oplus E_2 := E_1 \cup E_2 \cup \{(v_p, v) \,|\, v \in V\} \setminus \{(v, v_l), (v, v_r) \,|\, v \in V\}.$$

This recursion proceeds until we have mapped the root biconnected component; then $c(\{e(r(G))\}, LCA(r(G)))$ is the cost of an MPR from G to S. The MPR itself can then be found through standard back-tracking. Formal pseudocode for this algorithm is given in Algorithm 1.

3.2 Proof of correctness

We now show formally that the algorithm works as claimed. We first require some results on the configurations (E, x) that are added to A.

Algorithm 1 Given a gene network G and a corresponding species tree S, return the cost of an MPR between G and S.

```
1:  for each reticulation node $v \in V(G)$ do
2:      calculate a set of possible mappings $M_v \subseteq V(S)$
3:  end for

4:  for each $v \in L(G)$ do
5:      $c(\{e(v)\}, s(v)) \leftarrow 0$
6:  end for

7:  for each biconnected component $B \subseteq V(G)$ with $B \not\subseteq L(G)$ in bottom-up order do
8:      if $B$ is tree-child then
9:          $c(\{e(r(B))\}, LCA(r(B))) \leftarrow$ cost of LCA-HCA mapping for nodes of $B$ + costs of descendant biconnected components
10:         goto line 7
11:     end if

12:     $A \leftarrow \emptyset$
13:     for each $x \in V(S)$ with $x \leq LCA(r(B))$ in bottom-up order do
14:         $E_x \leftarrow \{(u,v) \in out(B) \mid LCA(v) = x\}$
15:         if $x \in L(S)$ then
16:             $c(E_x, x) \leftarrow \sum_{e \in E_x} c(\{e\}, x)$
17:             add $(E_x, x)$ to $A$ (even if $E_x = \emptyset$)
18:         else
19:             for each $(E_1, x_l), (E_2, x_r) \in A$ do
20:                 $c((E_1 \oplus E_2) \cup E_x, x) \leftarrow c(E_1, x_l) + c(E_2, x_r) + (2|E_1 \oplus E_2| - |E_1| - |E_2|) \cdot \lambda + \sum_{e \in E_x} c(\{e\}, x)$
21:                 add $((E_1 \oplus E_2) \cup E_x, x)$ to $A$
22:             end for
23:         end if
24:         repeat
25:             while $\exists (E, x) \in A$ s.t. $(u, u_l), (u, u_r) \in E$, $u \in V(G)$ is a divergence node do
26:                 $E' \leftarrow \{(u_p, u)\} \cup E \setminus \{(u, u_l), (u, u_r)\}$
27:                 $c(E', x) \leftarrow \min(c(E', x), c(E, x) + \delta)$
28:                 add $(E', x)$ to $A$
29:                 remove $(E, x)$ from $A$
30:             end while
31:             if $\exists (E, x) \in A$ s.t. $(u, u_l) \in E$ is unmarked, $u \in V(G)$ is a reticulation node then
32:                 $E' \leftarrow \{(u_p, u), (u_q, u)\} \cup E \setminus \{(u, u_l)\}$
33:                 $c(E', x) \leftarrow \min(c(E', x), c(E, x))$
34:                 add $(E', x)$ to $A$
35:                 if $x = \max(M_u)$ then
36:                     remove $(E, x)$ from $A$
37:                 else
38:                     mark $(u, u_l)$ in $(E, x)$
39:                 end if
40:             end if
41:         until $A$ does not change
42:     end for
43: end for

44: return $c(\{e(r(G))\}, LCA(r(G)))$
```

▶ **Lemma 8.** *If $(E, x) \in A$ at any stage of Algorithm 1, then:*

1. *for all $(u, v) \in E$, $LCA(v) \le x$;*
2. *E is either $\emptyset$ or a cut set of the gene network G;*
3. *E separates $r(B)$ and all nodes $v \in V(G)$ such that $(u, v) \in out(B)$ and $LCA(v) \le x$;*
4. *edges in E are non-comparable.*

Proof. We show this by induction on the elements of A in the order they are added. Consider an arbitrary biconnected component B. The initial elements of A are (E_x, x) as defined on line 14 for all $x \in L(S)$. If $E_x = \emptyset$, clearly all properties but the third are satisfied, and by definition there are no nodes such that $(u, v) \in out(B)$ and $LCA(v) \le x$, so property 3 is also satisfied. Otherwise, E_x consists of a number of out-edges (u, v) of B with $LCA(v) = x$. By virtue of the fact that B is a biconnected component, any set of its out-edges forms a cut set of G, so E is a cut set of G that separates $r(B)$ and all v such that $(u, v) \in out(B)$ and $LCA(v) \le x$. Thus the first 3 properties hold, and since out-edges of B are non-comparable, property 4 also holds.

Now suppose that the lemma holds for all $(E, x) \in A$ added up to some point. There are two possibilities for adding a new (E', x) to A, for the same species vertex x. The first one (line 28) has E' identical to E except for the removal of two edges (u, u_l) and (u, u_r), and the addition of the edge (u_p, u) for some divergence node u. Since $LCA(u_l), LCA(u_r) \le x$, then $LCA(u) \le x$, so property 1 holds. Clearly if E is a cut set of G, then E' is as well; furthermore, E' separates the same out-edges of B from $r(B)$ as E does, so properties 2 and 3 hold. Lastly, if an edge in E' is comparable to (u_p, u), then it must be comparable to either (u, u_l) or (u, u_r), causing a contradiction, so property 4 holds. Hence the lemma holds for (E', x).

The second possibility (line 34) has E' identical to E except for the removal of an edge (u, u_l) and the addition of the edges (u_p, u) and (u_q, u) for some reticulation node u. Again, we have $LCA(u) = LCA(u_l) \le x$, and it is clear that if E is a cut set of G, then E' is as well, and E' separates the same out-edges of B from $r(B)$. Finally, if an edge in E' is comparable to (u_p, u) or (u_q, u), then it is comparable to (u, u_l). Thus the lemma holds for (E', x).

Finally, if $(E_1, x_l), (E_2, x_r) \in A$, then $((E_1 \oplus E_2) \cup E_x, x)$ is added to A (line 21). All edges (u, v) in E_1, E_2, or E_x have $LCA(v) \le x$ from the induction assumption, and $E_1 \oplus E_2$ is derived from $E_1 \cup E_2$ by a series of operations similar to line 26. By similar reasoning to above, property 1 holds for all edges in $(E_1 \oplus E_2) \cup E_x$.

By the induction assumption, E_1 and E_2 are cut sets of G, and we have shown above that E_x is also a cut set. By construction, $E_1 \cup E_2$ is a cut set, and from similar reasoning to above, $E_1 \oplus E_2$ is a cut set, and thus so is $(E_1 \oplus E_2) \cup E_x$, so property 2 holds.

Furthermore, E_1 separates $r(B)$ and all nodes v such that $(u, v) \in out(B)$ and $LCA(v) \le x_l$, E_2 separates $r(B)$ and all nodes v such that $(u, v) \in out(B)$ and $LCA(v) \le x_r$, and E_x separates $r(B)$ and all nodes v such that $(u, v) \in out(B)$ and $LCA(v) = x$. Hence $(E_1 \oplus E_2) \cup E_x$ separates $r(B)$ and all nodes v such that $(u, v) \in out(B)$ and $LCA(v) \le x$, so property 3 holds.

Lastly, E_1 only contains edges (u, v) such that $LCA(v) \le x_l$, and E_2 only contains edges (u, v) such that $LCA(v) \le x_r$. Since E_x only contains out-edges (u, v) of B with $LCA(v) = x$, edges in E_x are non-comparable to edges in E_1 and E_2. Hence all edges in $(E_1 \oplus E_2) \cup E_x$ are non-comparable, and property 4 holds.

Therefore the lemma holds for $((E_1 \oplus E_2) \cup E_x, x)$, and the result follows by induction. ◀

Note that if E is a cut set of G, then the gene subnetwork forest G_E is well-defined, and thus so is the cost $c(E, x)$. We define the terminology "(f, ϵ) maps E to x" as meaning that for all $(u, v) \in E$, $f(v) \leq x$, and either $f(u) \geq x$ for $\epsilon(u) \neq \mathbb{S}$ or $f(u) > x$ for $\epsilon(u) = \mathbb{S}$. We can consider that each configuration (E, x) in the active set A represents all (f, ϵ) that map E to x.

▶ **Theorem 9.** *Algorithm 1 returns the cost of an MPR between G and S.*

Proof. Consider an arbitrary non-leaf biconnected component B (line 7). We claim that after each species vertex x is processed (line 13), then if $(E, x) \in A$, $c(E, x)$ is set to the minimal cost to reconcile the gene subnetwork forest G_E to the species subtree S_x.

First consider a version of Algorithm 1 with no configurations removed from A (lines 29 and 36), and moreover at line 21 also adds all configurations (E', x) for all $V' \subseteq V$, where we define V as in Definition 7, and

$$E' := E_1 \cup E_2 \setminus \{(v, v_l), (v, v_r) \mid v \in V'\} \cup \{(v_p, v) \mid v \in V'\} \cup E_x.$$

For each configuration $(E, x) \in A$, this version of Algorithm 1 simply considers all possible events (divergences, reticulations, and speciations) that could happen immediately above E, and adds the resulting configuration to A. It is thus a standard dynamic programming algorithm which returns the cost of an MPR between G and S.

To show that Algorithm 1 as written also returns the cost of an MPR, we need to show that the configurations that are removed from (or not added to) A cannot contribute to a most parsimonious reconciliation, other than to lead to configurations that have already been added to A. In other words, we need to show the following three lemmas:

▶ **Lemma 10.** *If (f, ϵ) is an MPR between G and S that maps E to x, and $(u, u_l), (u, u_r) \in E$ for a divergence node $u \in V(G)$, then $f(u) = x$ and $\epsilon(u) = \mathbb{D}$.*

Proof. Suppose that $f(u) = y > x$. Consider the reconciliation (f', ϵ) where $f'(u) = x$ and $f'(v) = f(v)$ for all other $v \in V(G)$. By assumption, $f'(u_l), f'(u_r) \leq x$, so this is a valid reconciliation. Also, since $f(u) > x \geq LCA(u)$, we know that $\epsilon(u) = \mathbb{D}$.

Now (f, ϵ) has $2d(y, x)$ more losses than (f', ϵ) on the the branches (u, u_l) and (u, u_r), and $d(y, x)$ fewer losses on the branch (u_p, u). Hence $c(f, \epsilon) > c(f', \epsilon)$, a contradiction. Therefore $f(u) = x$. If $\epsilon(u) = \mathbb{S}$, then we require $f(u) > x$, so $\epsilon(u) = \mathbb{D}$. ◀

▶ **Lemma 11.** *If (f, ϵ) is an MPR between G and S that maps E to x, $(u, u_l) \in E$ for a reticulation node $u \in V(G)$, and $x = max(M_u)$, then $f(u) = x$.*

Proof. By assumption, we must have $f(u) \geq x$, but by definition of M_u, $f(u) \in M_u$. Hence $f(u) = x$. ◀

▶ **Lemma 12.** *If (f, ϵ) is an MPR between G and S that maps E_1 to x_l and E_2 to x_r, and $\{(u, u_l), (u, u_r)\} \in \{E_1, E_2\}$ for a divergence node $u \in V(G)$, then $f(u) = x$ and $\epsilon(u) = \mathbb{S}$.*

Proof. Suppose that $f(u) = y \geq x$ and $\epsilon(u) = \mathbb{D}$. Consider the reconciliation (f', ϵ') where $f'(u) = x$ and $\epsilon'(u) = \mathbb{S}$, and $f'(v) = f(v)$, $\epsilon'(v) = \epsilon(v)$ for all other $v \in V(G)$. By assumption, $f'(u_l), f'(u_r) < x$, so this is a valid reconciliation.

Now (f', ϵ') has one fewer duplication than (f, ϵ) and $2(d(y, x) + 1)$ fewer losses on the branches (u, u_l) and (u, u_r). Conversely, it has $d(y, x)$ more losses on the branch (u_p, u). Hence $c(f, \epsilon) > c(f', \epsilon)$, a contradiction. Hence $\epsilon(u) = \mathbb{S}$, which forces $f(u) = x$. ◀

These three lemmas show that the configurations that are removed from A do not eliminate any reconciliations which are most parsimonious. Hence Algorithm 1 returns the cost of an MPR between G and S. ◀

3.3 Efficiency

We now show that the algorithm is exponential in the level of the gene network. Although it has been shown [1] that any algorithm that is fixed-parameter tractable in the number of reticulations is also fixed-parameter tractable in the level of the network, the level is typically a much smaller number in practice.

▶ **Theorem 13.** *Algorithm 1 runs in $O(|G||S|2^k)$ time, where k is the level of the network G.*

Proof. Consider an arbitrary non-leaf biconnected component B with r reticulation nodes. After each species vertex x is processed (line 13), from Lemma 8, if $(E, x) \in A$, then E is a cut set of G which separates $r(B)$ and all out-edges (u, v) of B where $LCA(v) \leq x$.

We claim that E is completely determined by which reticulation nodes in B it separates from $r(B)$. For a given E, let R be the set of reticulation nodes which E does not separate from $r(B)$. (Note that R may be empty, for example if B has no reticulation nodes.) Let E' be the set $\{(u, v) \,|\, u \in B, LCA(v) \leq x, v \ngeq R\}$, where $v \ngeq R$ means that there exists no $r \in R$ where $v \geq r$. We will show that E is the set of maximal edges of E'.

To show this, assume that this is not the case. By construction, E' contains all possible edges that could be in E, under the conditions that E is mapped to x and does not separate R from $r(B)$, and so $E \subseteq E'$. Therefore, there must be an edge in E which is not maximal in E'. Let e be such an edge which is minimal according to its parent node, i.e., $e = (u, v) \in E$ and there is no edge $(u', v') \in E$ which is not maximal in E' with $u > u'$. There are now two possibilities:

- u is a reticulation node. Then either $u \in R$, in which case e is maximal in E' (a contradiction), or $u \notin R$, in which case there must exist an edge in E separating $r(B)$ from u. This edge is comparable to e, a contradiction.
- u is a divergence node. We must have $LCA(u) \leq x$, as otherwise e would be maximal in E'. Now consider the sibling edge of (u, v), which we denote (u, v'). By assumption that e is minimal according to its parent node, no strict descendants of (u, v') are in E. Similarly, no ancestors of (u, v') are in E, as they are comparable to e. However, $LCA(v') \leq LCA(u) \leq x$, so (u, v') is the ancestor of at least one out-edge (u'', v'') of B where $LCA(v'') \leq x$. From Lemma 8, we know that E separates $r(B)$ and this edge, which means that $(u, v') \in E$. But since both (u, v) and (u, v') are in E, we can contract the divergence u and remove (E, x) from A, a contradiction.

Hence E is the set of maximal edges of E', and the claim is proved. Since there are r reticulation nodes in B, there are at most 2^r possible choices of R, and thus at most 2^r possible choices for E for a given x.

Now, there are at most $|G|$ biconnected components, for each of which we process $|S|$ species vertices, at each of which we calculate $c(E, x)$ for at most 2^k sets, where k is the level of G. The result follows. ◀

4 Simulation results

To test the algorithm, we perform some simulations. We first simulate a species tree of m species using a Yule (pure-birth) process with a birth rate of 1. Then we simulate a gene network within the species tree using a birth-and-death process as follows. We begin with a single gene lineage at the root of the species tree. At any time, a gene lineage may duplicate (independently of other lineages) at a constant rate of D (events per unit time), or be lost at a constant rate of L. If the last gene lineage is lost, we re-simulate the gene network from the beginning. In addition, if there are $k > 1$ gene lineages in a species, then at a constant rate of $R(k-1)$, two lineages from that species are randomly selected to recombine, with a breakpoint chosen uniformly at random from the sequence interval, taken to be $[0, 1]$. When a species speciates, all gene lineages contained in that species diverge into both child species.

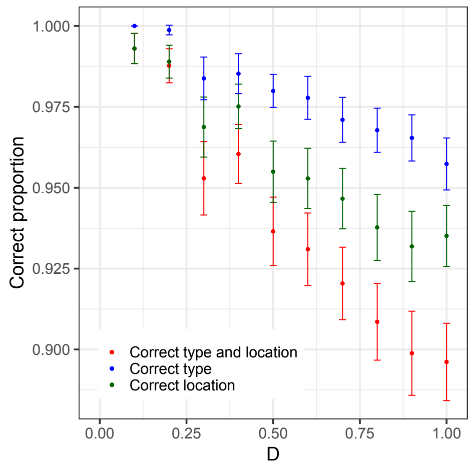
After simulating the gene network, we then reconcile it to the species tree with costs of $\delta = \lambda = 1$. (If the algorithm takes longer than 10 minutes, we terminate it to keep simulations tractable; this occurred in less than 5% of cases at any parameter setting, and usually not at all.) We then compare the reconciled gene network to the true history by considering each event type in turn, examining all events of that type in the true reconciliation, and determining the proportions of these events that are:

- assigned the correct event type (reticulations are always assigned the correct event type);
- correctly placed in the species tree;
- both assigned the correct event type and correctly placed in the species tree.

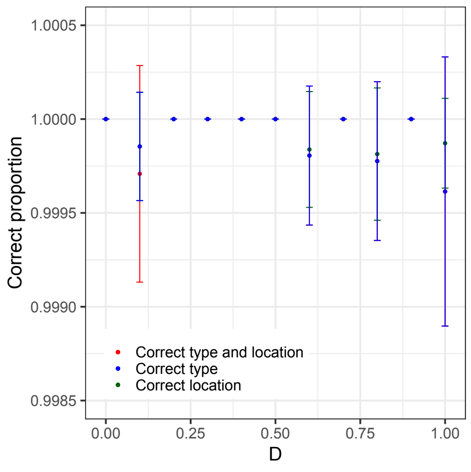
In addition, for each pair of extant genes, we determine the paralogy/orthology relationship between the pair by tracing their LCAs in the gene network. Due to recombinations, there may be more than one LCA, but there is always exactly one LCA for any point in the sequence interval. If an LCA is a duplication event, the pair is considered to be paralogous at that point in the sequence, and if it is a speciation, the pair is considered to be orthologous. We calculate the segments of paralogy/orthology for each gene pair and determine the proportion of the total sequence length for which this relationship is correctly inferred.

We first study the case where the duplication and loss rates vary, but are constrained to be equal, and the recombination rate is held fixed. Figure 4 shows the results for the values $m = 10$, $D = L \in [0, 1]$, $R = 0.5$, and 1000 replicates for each parameter setting. (Although the species tree is quite small, the ability to have both duplication and recombination events means that the gene network space is unbounded, and indeed the largest gene network in these simulations contains 123 leaves and 161 events.) We see that in all cases, all types of events are very accurately inferred, with over 90% of duplication events inferred with the correct type and placed in the correct location in the species tree; likewise for reticulation events (by definition, they are always inferred with the correct type). Speciation events are inferred even more precisely, with virtually all of them correctly inferred. As expected, accuracy decreases as the duplication and loss rate increase. Likewise, paralogy/orthology relationships are very accurately inferred; on average, the correct relationship is inferred for over 96% of the sequence interval, with accuracy again decreasing with increasing duplication/loss rate.

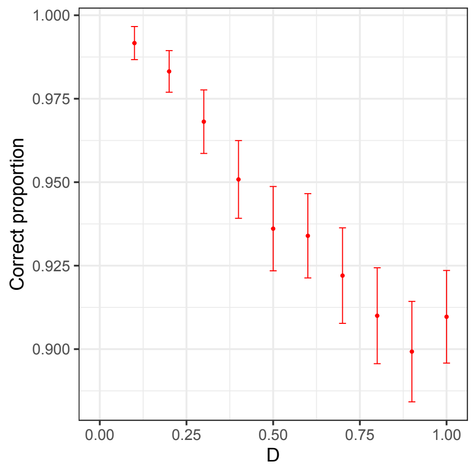
Next, we study the case where the duplication and recombination rates vary, but are constrained to be equal, and the loss rate is held fixed. Figure 5 shows the results for $m = 10$, $D = R \in [0, 1]$, $L = 0.5$, and 1000 replicates for each parameter setting. As above, all events are very accurately inferred (over 90% in all cases), as are the paralogy/orthology relationships (over 98% in all cases). Interestingly, accuracy increases rather than decreases for increasing duplication/recombination rate. A possible explanation for this is that while losses induce a topological discordance between the gene network and species tree, this is less true for recombinations, since both lineages that recombine are still present in the gene network. Thus, increasing the loss rate results in decreasing accuracy, but the same is not true for the recombination rate.



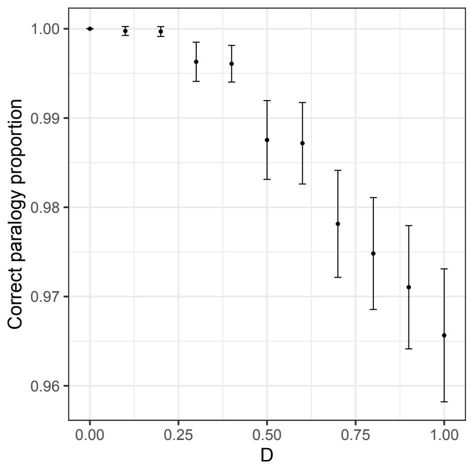
(a) Duplication events.



(b) Speciation events.

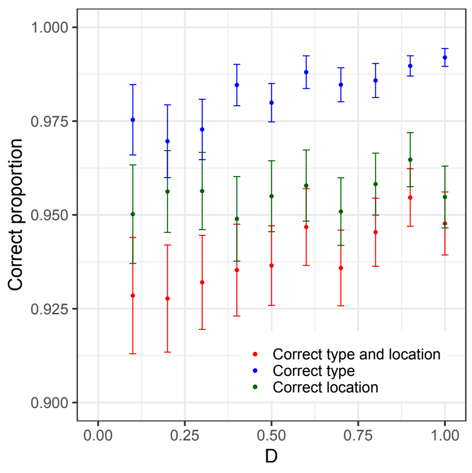


(c) Reticulation events.

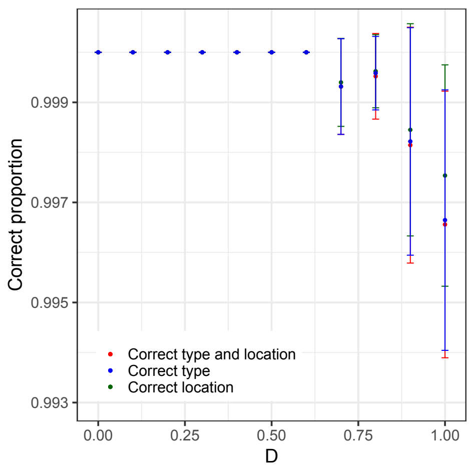


(d) Paralogy.

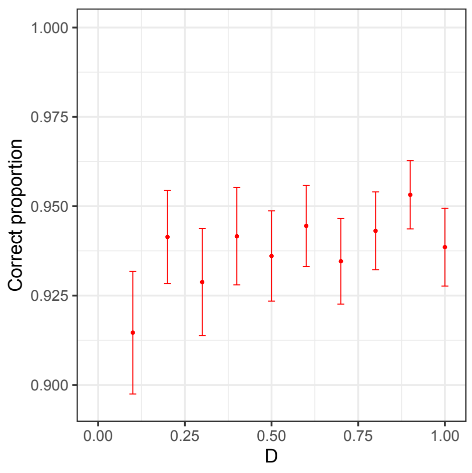
Figure 4 The average proportion of (a)–(c) correctly inferred events and (d) correctly inferred paralogy/orthology for equal duplication and loss rates. Error bars are twice the standard error for 1000 replicates.



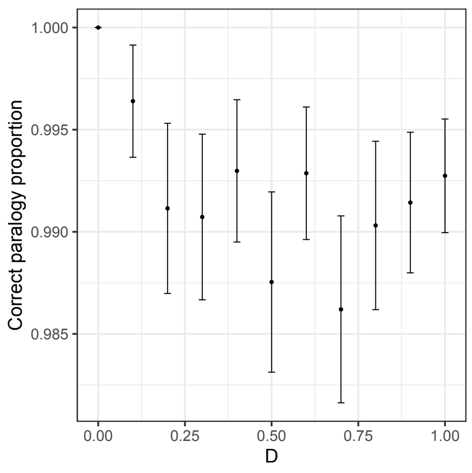
(a) Duplication events.



(b) Speciation events.



(c) Reticulation events.



(d) Paralogy.

Figure 5 The average proportion of (a)–(c) correctly inferred events and (d) correctly inferred paralogy/orthology for equal duplication and recombination rates. Error bars are twice the standard error for 1000 replicates.

5 Discussion

In this paper, we re-examine the problem of the reconciliation of a gene network to a species tree. We devise an algorithm that finds the most parsimonious reconciliation for a general gene network, and show that it is exponential in the level of the network. Simulations show that this is a practical and efficient algorithm to solve the network-tree reconciliation problem.

The benefit of this approach is that it allows a fine-grained comparison of paralogy/orthology relationships between genes. By acknowledging the process of paralog exchange, we now see that it is possible for genes to be partially paralogous and partially orthologous along their sequences. Our reconciliation algorithm allows us to recover these relationships and thus study their functional implications at a deeper level.

The paradigm used in our algorithm is different from the traditional dynamic programming approaches used for e.g., DTL reconciliations; instead of recursing on the branches of the gene tree, we recurse on the branches of the species tree. The former approach, while used widely, fails in many models where gene lineages are dependent. In addition to the gene network model, this also includes many of the models mentioned in the Introduction, such as segmental duplications or incomplete lineage sorting. It has been proven that many of these models are NP-hard, although the difficulty of the gene network problem itself is an open question. Our work here provides another avenue to approach these problems in a tractable manner.

Our algorithm takes costs δ and λ for duplications and losses as input, but does not specify how to choose them. This is a common issue for parsimonious reconciliation models, where one potential approach is to explore the space of these costs to determine Pareto-optimal reconciliations [18]. These reconciliations are typically computed using the traditional dynamic programming framework, so further work is needed to calculate them for the network-tree problem.

There still remains the problem of how to reconstruct the gene network in the first place. Phylogenetic network reconstruction methods are still relatively inefficient, and more pertinently aim to reconstruct a species network, rather than a gene network. ARG reconstruction methods seem more likely, and recent dramatic advances in their efficiency allow the inference of such gene networks at scale. However, more must be done in order to infer them across larger evolutionary distances and in particular over large numbers of species. Our reconciliation method may aid in this direction by allowing a species tree-aware method to relate (and thus score) a prospective gene network to the species tree.

Finally, a logical extension of this method is to the case where both gene and species phylogenies are in the form of networks, which is the subject of future work.

Data availability

All code used in the simulations is available at
`https://github.com/yao-ban/network-tree-reconciliation`.

References

1 Magnus Bordewich, Celine Scornavacca, Nihan Tokac, and Mathias Weller. On the fixed parameter tractability of agreement-based phylogenetic distances. *Journal of Mathematical Biology*, 74:239–257, 2017.

2 Débora YC Brandt, Christian D Huber, Charleston WK Chiang, and Diego Ortega-Del Vecchyo. The promise of inferring the past using the ancestral recombination graph. *Genome Biology and Evolution*, 16(2):evae005, 2024.

3 Gabriel Cardona, Francesc Rossello, and Gabriel Valiente. Comparison of tree-child phylogenetic networks. *IEEE/ACM Transactions on Computional Biology and Bioinformatics*, 6(4):552–569, 2009.

4 Yao-ban Chan, Vincent Ranwez, and Céline Scornavacca. Inferring incomplete lineage sorting, duplications, transfers and losses with reconciliations. *Journal of Theoretical Biology*, 432:1–13, 2017.

5 Yao-ban Chan and Charles Robin. Reconciliation of a gene network and species tree. *Journal of Theoretical Biology*, 472:54–66, 2019.

6 Antoine Claessens, William L Hamilton, Mihir Kekre, Thomas D Otto, Adnan Faizullabhoy, Julian C Rayner, and Dominic Kwiatkowski. Generation of antigenic diversity in *Plasmodium falciparum* by structured rearrangement of *Var* genes during mitosis. *PLoS Genetics*, 10(12):e1004812, 2014.

7 Riccardo Dondi, Manuel Lafond, and Celine Scornavacca. Reconciling multiple genes trees via segmental duplications and losses. *Algorithms for Molecular Biology*, 14:1–19, 2019.

8 Jean-Philippe Doyon, Celine Scornavacca, K Yu Gorbunov, Gergely J Szöllősi, Vincent Ranwez, and Vincent Berry. An efficient algorithm for gene/species trees parsimonious reconciliation with losses, duplications and transfers. In *RECOMB International Workshop on Comparative Genomics*, pages 93–108. Springer, 2010.

9 Benjamin Drinkwater, Angela Qiao, and Michael A Charleston. WiSPA: A new approach for dealing with widespread parasitism. *arXiv preprint arXiv:1603.09415*, 2016.

10 Morris Goodman, John Czelusniak, G William Moore, Alejo E Romero-Herrera, and Genji Matsuda. Fitting the gene lineage into its species lineage, a parsimony strategy illustrated by cladograms constructed from globin sequences. *Systematic Biology*, 28(2):132–163, 1979.

11 Robert C Griffiths and Paul Marjoram. Ancestral inference from samples of DNA sequences with recombination. *Journal of Computational Biology*, 3(4):479–502, 1996.

12 Damir Hasić and Eric Tannier. Gene tree reconciliation including transfers with replacement is NP-hard and FPT. *Journal of Combinatorial Optimisation*, 38(2):502–544, 2019.

13 Damir Hasić and Eric Tannier. Gene tree species tree reconciliation with gene conversion. *Journal of Mathematical Biology*, 78(6):1981–2014, 2019.

14 Melissa J Hubisz, Amy L Williams, and Adam Siepel. Mapping gene flow between ancient hominins through demography-aware inference of the ancestral recombination graph. *PLoS Genetics*, 16(8):e1008895, 2020.

15 Daniel H Huson, Regula Rupp, and Celine Scornavacca. *Phylogenetic networks: concepts, algorithms and applications*. Cambridge University Press, 2010.

16 Alexander L Lewanski, Michael C Grundler, and Gideon S Bradburd. The era of the ARG: An introduction to ancestral recombination graphs and their significance in empirical evolutionary genomics. *PLoS Genetics*, 20(1):e1011110, 2024.

17 Qiuyi Li, Celine Scornavacca, Nicolas Galtier, and Yao-ban Chan. The multilocus multispecies coalescent: a flexible new model of gene family evolution. *Systematic Biology*, 2020.

18 Ran Libeskind-Hadas, Yi-Chieh Wu, Mukul S Bansal, and Manolis Kellis. Pareto-optimal phylogenetic tree reconciliation. *Bioinformatics*, 30(12):i87–i95, 2014.

19 Hugo Menet, Vincent Daubin, and Eric Tannier. Phylogenetic reconciliation. *PLoS Computational Biology*, 18(11):e1010621, 2022.

20 Matthew D Rasmussen and Manolis Kellis. Unified modeling of gene duplication, loss, and coalescence using a locus tree. *Genome Research*, 22(4):755–765, 2012.

21 Celine Scornavacca, Joan Carles Pons Mayol, and Gabriel Cardona. Fast algorithm for the reconciliation of gene trees and LGT networks. *Journal of Theoretical Biology*, 418:129–137, 2017.

22 Thu-Hien To and Celine Scornavacca. Efficient algorithms for reconciling gene trees and species networks via duplication and loss events. *BMC Genom.*, 16(10):S6, 2015.

23 Yi-Chieh Wu, Matthew D Rasmussen, Mukul S Bansal, and Manolis Kellis. Most parsimonious reconciliation in the presence of gene duplication, loss, and deep coalescence using labeled coalescent trees. *Genome Research*, 24(3):475–486, 2014.

24 Yufeng Wu. Coalescent-based species tree inference from gene tree topologies under incomplete lineage sorting by maximum likelihood. *Evolution*, 66(3):763–775, 2012.

25 Louxin Zhang. On a Mirkin-Muchnik-Smith conjecture for comparing molecular phylogenies. *Journal of Computational Biology*, 4(2):177–187, 1997.