JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 15, Number 8, 2008

Pp. 981–1006
DOI: 10.1089/cmb.2008.0092

Reconciliation with Non-Binary Species Trees

BENJAMIN VERNOT,[1]* MAUREEN STOLZER,[1]* AITON GOLDMAN,[1]
and DANNIE DURAND[1,2]

ABSTRACT

Reconciliation extracts information from the topological incongruence between gene and species trees to infer duplications and losses in the history of a gene family. The inferred duplication-loss histories provide valuable information for a broad range of biological applications, including ortholog identification, estimating gene duplication times, and rooting and correcting gene trees. While reconciliation for binary trees is a tractable and well studied problem, there are no algorithms for reconciliation with non-binary species trees. Yet a striking proportion of species trees are non-binary. For example, 64% of branch points in the NCBI taxonomy have three or more children. When applied to non-binary species trees, current algorithms overestimate the number of duplications because they cannot distinguish between duplication and incomplete lineage sorting. We present the first algorithms for reconciling binary gene trees with non-binary species trees under a duplication-loss parsimony model. Our algorithms utilize an efficient mapping from gene to species trees to infer the minimum number of duplications in $O(|V_G| \cdot (k_S + h_S))$ time, where $|V_G|$ is the number of nodes in the gene tree, h_S is the height of the species tree and k_S is the size of its largest polytomy. We present a dynamic programming algorithm which also minimizes the total number of losses. Although this algorithm is exponential in the size of the largest polytomy, it performs well in practice for polytomies with outdegree of 12 or less. We also present a heuristic which estimates the minimal number of losses in polynomial time. In empirical tests, this algorithm finds an optimal loss history 99% of the time. Our algorithms have been implemented in NOTUNG, a robust, production quality, tree-fitting program, which provides a graphical user interface for exploratory analysis and also supports automated, high-throughput analysis of large data sets.

Key words: deep coalescence, gene duplication, gene loss, lineage sorting, non-binary species trees, polytomy, reconciliation.

1. INTRODUCTION

RECONCILIATION IS THE PROCESS of constructing a mapping between a gene family tree and a species tree in order to infer the history of gene duplications and losses during the evolution of the

*These authors contributed equally to this work.
[1]Department of Biological Sciences, Carnegie Mellon University, Pittsburgh, Pennsylvania.
[2]Department of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania.

gene family. Reconciliation is widely used for evolutionary applications in medicine, developmental biology and plant sciences. It is the most reliable approach for identifying orthologs for use in function prediction, gene annotation, planning experiments in model organisms, and identifying drug targets (Bourgon et al., 2004; Searls, 2003). Reconciliation is used to estimate times of duplication (Ermolaeva et al., 2003; Gu et al., 2002; Ruvinsky and Silver, 1997; Vandepoele et al., 2004; Hahn, 2007), to test whole genome duplication hypotheses (Vandepoele et al., 2003; Wang et al., 2005; McLysaght et al., 2002; Paterson et al., 2004) and to identify recently duplicated genes that may be sites of adaptation (Hahn et al., 2007). By correlating specific duplications with the emergence of novel cellular functions or morphological features, reconciliation provides clues to the functions of newly discovered genes (Demuth et al., 2006; Wheeler et al., 2001). Minimizing duplications and losses provides a basis for rooting an unrooted tree (Chen et al., 2000) and for selecting alternate gene or species tree topologies (Chen et al., 2000; Durand et al., 2006a; Bourgon et al., 2004; Nam and Masatoshi, 2005).

Reconciliation of binary trees is a well-studied problem (Guigó et al., 1996; Hallett and Lagergren, 2000; Ma et al., 2000; Mirkin et al., 1995; Page, 1994; Stege, 1999; Zhang, 1997; Eulenstein et al., 1998). A number of software packages for this problem are available (Page and Charleston, 1997; Page, 1998; Zmasek and Eddy, 2001b; Zmasek and Eddy, 2001a; Dufayard et al., 2005; Roth et al., 2005; Durand et al., 2006a; Berglund-Sonnhammer et al., 2006; Sennblad et al., 2007), as are high throughput reconciliation tools for automated processing of databases of molecular phylogenies (Perrière et al., 2000; Dufayard et al., 2005; Roth et al., 2005; Li et al., 2006) and analysis of genome-scale data sets (Hahn et al., 2007; Blomme et al., 2006; Demuth et al., 2006). Reconciliation is also the kernel of a related, but more complex, problem: inferring a species tree from many gene trees (Ma et al., 2000, and work cited therein). Historically, most work involving reconciliation has focused on duplications; however, losses are also an important factor in gene family evolution. Work that acknowledges this importance and incorporates loss information is emerging (Chauve et al., 2007; Chen et al., 2000; Guigó et al., 1996)

Reconciliation relies on the observation that discordance between a binary gene tree and a binary species tree is evidence that genes diverged through processes other than speciation. These processes include gene duplication and loss, incomplete lineage sorting, and horizontal gene transfer. Previous studies on binary reconciliation explain tree disagreement exclusively in terms of gene duplication and loss. A few methods that consider horizontal gene transfer have been proposed (Gorecki, 2004; Hallett and Lagergren, 2001; Hallett et al., 2004). However, to our knowledge, there are none that consider incomplete lineage sorting in binary trees. Since the probability of incomplete lineage sorting decreases as time between speciation events increases (Pamilo and Nei, 1988; Takahata and Nei, 1985; Maddison, 1997; Tajima, 1983; Hudson, 1990), ignoring incomplete lineage sorting as a cause of discordance is justified if the branch lengths in the species tree are sufficiently long.

In contrast, when the species tree is non-binary, incomplete lineage sorting is a significant phenomenon that cannot be ignored (Pollard et al., 2006). Since duplication and incomplete lineage sorting have different consequences for the interpretation of phylogenetic studies, it is essential to distinguish between discordances that can only be explained by duplication (*required* duplications) and discordances that could be due to either duplication or incomplete lineage sorting (*conditional* duplications). As we demonstrate below, standard binary reconciliation cannot make this distinction.

As the tree of life project gains momentum, it is becoming evident that reconciliation with non-binary species trees is not a minor problem relegated to a few obscure species lineages. Rather, 64% of branch points in the NCBI taxonomy (Wheeler et al., 2005), one of the most widely used databases of species phylogenies, have more than two children. A number of well-documented analyses of simultaneous divergences have been reported (Jackman et al., 1999; Poe and Chubb, 2004; Melnick et al., 1993; Hoelzer and Melnick, 1994; Salzburger et al., 2002). Considering the large number of non-binary species trees, reconciliation methods for non-binary trees are urgently needed.

Our contributions: To address this need, we present novel algorithms to reconcile a rooted binary gene tree, $T_G = (V_G, E_G)$, with a rooted non-binary species tree, $T_S = (V_S, E_S)$. Under the assumption of duplication-loss parsimony, our algorithms infer (1) the number of duplications and losses that occurred, (2) the species in which those events occurred, and (3) the gene tree lineage in which the events occurred. From these, the total number of duplication, speciation and loss events, as well as the temporal order of those events in each lineage, can be determined. Our algorithms consider only discordance due to duplication and incomplete lineage sorting. We further assume, like previous authors, that the probability of incomplete lineage sorting is negligible when a node in the species tree

is binary. Incomplete lineage sorting in binary species trees and horizontal gene transfer are reserved for future work.

We first present a mapping from nodes in V_G to sets of nodes in V_S that allows us to test efficiently whether a discordance at a given node is a conditional or required duplication. The maximum size of the set labeling any node in T_G is $O(k_S)$, where k_S is the maximum outdegree in T_S. Using this mapping, our algorithm infers all conditional and required duplications in $O(|V_G| \cdot (k_S + h_S))$ time, where h_S is the height of T_S. Each inferred duplication is assigned to a node in T_G, indicating the species in which it occurred and the timing of the duplication relative to other events in the gene family history.

We also present parsimony algorithms to infer the minimum number of gene losses. The timing of each inferred loss is indicated by assigning a node representing the loss to an edge in E_G. This loss node is labeled with the species in which the loss occurred. For binary species trees, the edge to which each loss is assigned is unambiguously determined by the reconciliation. In contrast, for a loss associated with a polytomy in a non-binary species tree, it is not generally possible to determine the exact lineage in the gene tree in which the loss occurred. In such cases, the loss is associated with several possible edges in E_G, corresponding to alternate hypotheses regarding when the loss occurred. Parsimony provides a principled basis for reducing this uncertainty: the number of alternate hypotheses may be reduced by assigning losses to edges in E_G such that the total number of losses is minimized.

Two considerations influence the total number of losses associated with a particular assignment of losses to edges. First, the position of the loss relative to duplications in T_G influences the total number of losses. Assigning a loss to an edge above a duplication in T_G implies that the loss occurred before the duplication. In this case, only one loss is inferred. Assigning the loss below that duplication implies that the duplication occurred first. In this case, two losses must be inferred, one for each duplicated copy. Second, under certain circumstances, losses in sibling species can be explained by a single loss in their common ancestor. We can reduce the number of losses by selecting assignments that maximize the opportunities to combine losses that share a parent. As we show later in the paper, these considerations are not independent of one another. While assigning a loss below a duplication usually increases the total number of losses, in some cases the duplicated losses can be combined with other losses that occurred after the duplication, resulting in fewer total losses.

We present an exact algorithm that infers a history with the fewest losses, while taking both of the above considerations into account. Despite the exponential complexity of this algorithm ($O(|V_G|k_S 2^{2k_S})$), in empirical tests our exact algorithm reconciled 1174 trees derived from the TreeFam (Li et al., 2006) database in about two and a half minutes. We also present a heuristic that runs in polynomial time ($O(|V_G| \cdot (k_S + h_S))$). This heuristic reconciled the same 1174 trees in 48 seconds. This heuristic places losses as close to the root as possible to avoid duplicating losses. Once all losses have been assigned to edges, losses are combined where possible. In a comparison of the two methods on these 1174 trees, the heuristic found the optimal solution in more than 99% of the cases studied. Out of the seven trees in which the heuristic did not find the optimal solution, in the worst case, the number of losses was overestimated by four losses out of a total of 249.

Our algorithms have been implemented in NOTUNG, a software package that takes trees in the widely used Newick format as input, permitting interoperability with a wide range of phylogeny reconstruction and drawing packages. In addition to reconciling binary gene trees with non-binary species trees, NOTUNG can reconcile both binary and non-binary gene trees with binary species trees (Durand et al., 2006b). NOTUNG can also apply duplication-loss parsimony to reduce uncertainty in a gene tree with weakly supported edges, to root a gene tree, and to resolve polytomies in a gene tree (Chen et al., 2000; Durand et al., 2006a). Our software is implemented in Java and runs on Windows, Unix and Mac OS X. It is freely available at *www.cs.cmu.edu/∼durand/Notung*.

Roadmap: In the next section, we introduce notation and review the standard algorithm for reconciliation of binary trees. In Section 3, we review the relevant models for non-binary gene and species trees in the molecular evolution literature and give formal definitions for required and conditional duplications based on this foundation. Next we present our non-binary reconciliation algorithms. Duplications are discussed in Section 4. In Section 5, we discuss losses and present an exact algorithm and a heuristic for inferring the minimum number of losses, as well as conditional and required duplications. We discuss related work by other authors in Section 6. In Section 7, we demonstrate the utility of our methods with analyses of real data sets using our software. In the conclusion, we discuss probabilistic approaches to reconciliation and describe directions for future work.

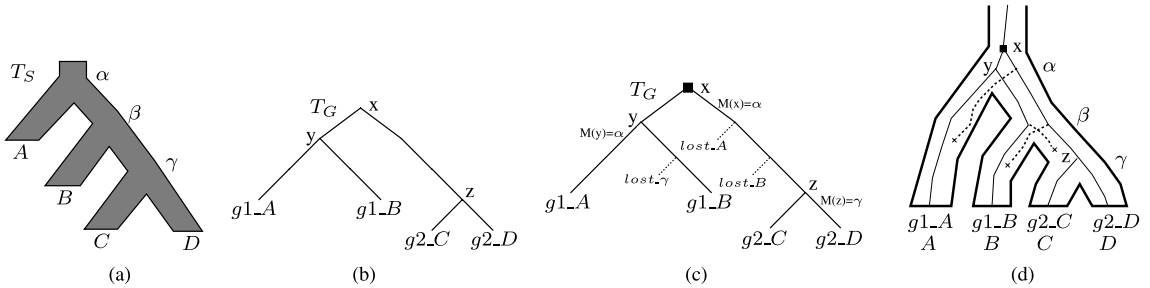

FIG. 1. LCA reconciliation. **(a)** A binary species tree. **(b)** A hypothetical binary gene tree with genes sampled from species in (a). **(c)** The gene tree (b) reconciled with species tree (a). **(d)** The gene tree (b) embedded in the species tree (a). The black squares indicate duplications, and dotted lines indicate losses.

2. NOTATION AND BINARY RECONCILIATION

In this section, we introduce notation and review the standard reconciliation algorithm for binary trees. The trees shown in Figure 1 will be used throughout this section to exemplify notation. In tree figures, the label g_s denotes a gene that is sampled from species s.

Let $T_i = (V_i, E_i)$ be a rooted tree, where V_i is the set of nodes in T_i, and E_i is the set of edges. $L(T_i)$ is the leaf set of T_i and $L(v)$ refers to the leaf set of a subtree rooted at $v \in V_i$. The root node of T_i is denoted as $root(T_i)$. $C(v)$ and $p(v)$ refer to the children and parent of v, respectively. If v is binary, $r(v)$ and $l(v)$ denote the right and left children of v. For example, in Figure 1, $p(y) = x$, and $C(y) = \{g1\_A, g1\_B\}$, where $l(y) = g1\_A$ and $r(y) = g1\_B$. A non-binary node in a tree is referred to as a *polytomy*. A *monophyletic* group is a set of nodes consisting of a node and all of its descendants; for example, in Figure 1a, $\{\gamma, C, D\}$ forms a monophyletic group. The expression $u \geq_i v$ indicates that for $u \in V_i$, either u is v, or u lies on the path from v to $root(T_i)$. In Figure 1c, $root(T_G) = x$ and $y \geq_G g1\_A$. We follow the computer science convention, in which the root is at the top of the tree, the leaves are at the bottom, and $p(g)$ is above g.

Reconciliation infers gene duplications and losses by fitting a gene tree to a species tree (Goodman et al., 1979; Page and Holmes, 1998; Ronquist, 2002). To perform this comparison for binary trees, a mapping between the gene tree and species tree is required. Let T_G be a binary gene tree and T_S be a binary species tree such that the genes in $L(T_G)$ were sampled from the species in $L(T_S)$. A mapping $M(\cdot)$ is constructed from each node $g \in V_G$ to a target node $s \in V_S$. If g is a leaf node, $M(g)$ is the species from which sequence g was sampled. If g is an internal node, $M(g)$ is the least common ancestor (LCA) of the target nodes of its children:

$$M(g) = LCA(M(l(g)), M(r(g))). \tag{1}$$

In our example, $M(g1\_A) = A$, since it is a leaf; $M(x) = LCA(M(y), M(z)) = LCA(\alpha, \gamma) = \alpha$. From this mapping both gene duplications and gene losses can be inferred. We refer to this algorithm for calculating duplications and losses as LCA reconciliation in order to distinguish it from the new reconciliation algorithms proposed for non-binary species trees in the next section.

By convention, duplications are assigned to nodes in V_G and losses to edges in E_G. Assigning a duplication to node $g \in V_G$ not only specifies its location in T_G, but also its location in T_S, via the mapping $M(\cdot)$. An inferred duplication at g implies that the duplication occurred between $p(M(g))$ and $M(g)$. The two resulting copies were present in species $M(g)$, and for at least one child c of $M(g)$ (if $M(g) \notin L(T_S)$), each copy persisted[1] in at least one leaf (not necessarily the same leaf) of the subtree of

[1]Duplication followed by loss of all descendants of one copy would not leave this pattern, but parsimony reconciliation models only consider cases where there is some remaining evidence of duplication.

T_S rooted at c. If $M(g) \in L(T_S)$, then both copies persisted in $M(g)$. For losses, the species, s, in which the loss occurred must be inferred explicitly. Assigning a loss in s to edge $(p(g), g)$ indicates that g was present in both $M(p(g))$ and $M(g)$, and was lost on the path from s' to s, where s' is a species on the path from $M(g)$ to $M(p(g))$ (i.e., $M(g) <_S s' \leq_S M(p(g))$).

Gene Duplications: A duplication is inferred at node g if and only if the children of g map to the same lineage in T_S; that is, there is some leaf $s \in L(T_S)$ such that both $l(M(g))$ and $r(M(g))$ are on the path from s to $root(T_S)$. This condition is true *iff*

$$M(g) = M(l(g)) \vee M(g) = M(r(g)). \tag{2}$$

Every node in T_G that is not designated a duplication node is a speciation node.

When the gene tree is binary, Equation (2) is sufficient to determine whether $l(g)$ and $r(g)$ map to the same lineage in T_S. Note, however, that Equation (2) only explicitly tests whether two copies were present in $M(g)$, but not whether each gene copy persisted in at least one leaf of a subtree descending from $M(g)$. The assumption of complete lineage sorting guarantees that the latter must also be true. However, as we demonstrate in Section 3, under incomplete lineage sorting, Equation (2) is not sufficient to determine whether two or more children map to the same lineage in T_S.

As an example of inferring gene duplication, we consider Figure 1. Figure 1d shows a duplication at node $x \in T_G$, prior to the species divergence at α. A descendant of $l(x)$ persisted in species B, while a descendant of $r(x)$ persisted in species C and D; thus, both copies are represented in at least one leaf of the subtree rooted at β. The gene tree embedded in the species tree in Figure 1d shows both copies of the gene on the edge (α, β). Although only one copy of the family survived in each species, discordance between the species tree in Figure 1a and the gene tree in Figure 1b provides sufficient evidence to infer a duplication at x. Because both x and one of its children (y) map to α, Equation (2) correctly identifies the duplication x.

Gene Losses: Losses can also be reconstructed from the mapping, $M(\cdot)$. For each $e = (p(g), g)$, the comparison of $M(p(g))$ and $M(g)$ determines the losses assigned to e. If $p(g)$ is a speciation node, and no loss occurred, then $M(p(g))$ must be the parent of $M(g)$ in the species tree. If $p(g)$ is a duplication node and no losses occurred, then $p(g)$ and g map to the same node in T_S. If either one of these conditions fail, then one or more losses must be inferred. Both situations arise in the example in Figure 1. Since x is a duplication node, $M(z) \neq M(x)$ indicates losses in A and B on the edge from x to z. Node $p(g1\_B) = y$ is a speciation node, but $M(y) = \alpha$ is not the parent of $M(g1\_B) = B$ in T_S, indicating the absence of $g1$ from species C and D. Note that these two losses can be explained more parsimoniously by the loss of a single ancestral gene in the ancestral species, γ.

In general, given a 1-1 mapping between a set of contemporary species in which a gene copy is absent (relevant to a single edge in E_G) and the leaves of a subtree rooted at $s \in V_S$, it is more parsimonious to infer a single loss in s. Under this assumption, when $p(g)$ is a speciation node, we infer $depth(M(g)) - depth(M(p(g))) - 1$ losses on edge e, one for each ancestral species on the path from $M(g)$ to $M(p(g))$ in T_S. If $p(g)$ is a duplication, the number of inferred losses is $depth(M(g)) - depth(M(p(g)))$.

The species associated with the losses inferred on $e = (p(g), g)$ are determined by walking up the species tree from $M(g)$ to $M(p(g))$. For each ancestral node $s \in V_S$ between $M(g)$ and $M(p(g))$, a loss is inferred in $l(s)$ or $r(s)$, whichever is not represented on the path from g to $p(g)$ in the gene tree. If $p(g)$ is a duplication node, an additional loss is inferred. For example, consider the losses on edge (x, z) in Figure 1c. Node x is a duplication node and $M(x) = \alpha$, but $M(z) = \gamma$, indicating that $depth(\gamma) - depth(\alpha) = 2$ losses occurred between x and z.

The reconciliation methods presented here fulfill the requirements for reconciliation algorithms specified in Section 1. Each duplication is assigned to a node $g \in V_G$, indicating the timing of the duplication relative to speciation and duplication events in the same gene tree lineage. The mapping $M(g)$ indicates the species lineage in which the duplication occurred. Similarly, each loss is associated with a node $s \in V_S$ and an edge $(g, p(g)) \in E_G$, indicating that the loss occurred between $p(s)$ and s in the species tree and between $p(g)$ and g in the gene tree. The total number of events is determined by summing over the set of all nodes and edges for duplications and losses, respectively.

3. MODELS FOR NON-BINARY SPECIES TREES

LCA reconciliation is based on the assumption that disagreement between a gene tree and a species tree indicates that one or more gene duplications or losses must have occurred. In this section, we show that when the species tree is non-binary, this assumption is no longer warranted. First we review current theory concerning polytomies in gene and species trees.

A polytomy may represent the simultaneous divergence of all its children (a hard polytomy) (Maddison, 1989). It may also indicate that the true binary branching pattern is unknown (a soft polytomy) (Maddison, 1989). A soft polytomy indicates that sufficient data is not available, or there is not enough signal in the data to determine the true branching order. This may occur when a sequence of binary divisions proceeds in close succession and the time between these events is insufficient to accumulate informative variation.

In gene trees, polytomies are always soft. Since each lineage in a gene tree represents exactly one gene, the result of any divergence is exactly two descendant sequences. Thus, the true branching pattern in a gene family is always binary (Hudson, 1990), and a polytomy can only represent uncertainty in the true, underlying history. In the current work, we assume that the true binary gene tree is known and focus exclusively on algorithms for binary gene trees.

In contrast, polytomies in a species tree can be either hard or soft. A species tree represents the evolution of a population of organisms. In this context, simultaneous divergence of three or more lineages does occur. For example, simultaneous divergence can result from the isolation of subpopulations within a widespread species by sudden meteorological or geological events, or from rapid expansion of the population into open territory, resulting in reproductive isolation. Examples of well-documented, simultaneous divergences in nature include *Anolis* lizards (Jackman et al., 1999), modern birds in the order *Neoaves* (Poe and Chubb, 2004), macaque monkeys (Melnick et al., 1993; Hoelzer and Melnick, 1994), auklets (Walsh et al., 1999), and African cichlid fishes (Salzburger et al., 2002).

A binary gene tree can be consistent with a hard polytomy in the species tree (Lyons-Weiler and Milinkovitch, 1997; Maddison, 1997), as illustrated in Figure 2. Since a node in the species tree represents a population with genetic diversity, there can be multiple alleles at the locus of interest. Discordance between gene and species trees that results from allelic variation is referred to as incomplete lineage sorting. The bifurcation at time t_1 shows the formation of two alleles through mutation. A second bifurcation occurs at t_2 and results in three different alleles at this locus. Note that this binary branching pattern represents allelic variation at a single locus, not the formation of independent loci through duplication. The true divergence between any two genetic lineages corresponds to the point at which the differences in alleles arose, not the time of speciation. Divergence that occurs much earlier than the time of speciation is referred to as a *deep coalescence event*. Given a hard polytomy with k leaves, all binary branching patterns with k leaves are

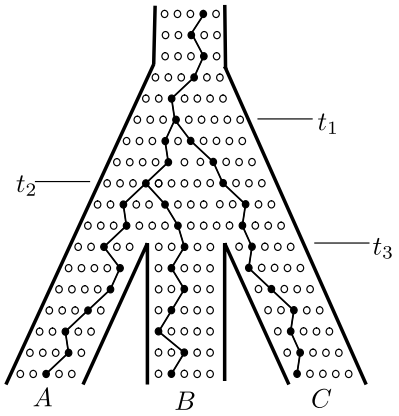

FIG. 2. Evolution of a single genetic locus in the context of a population. Each row represents a generation of individuals in the population at a specific point in time.

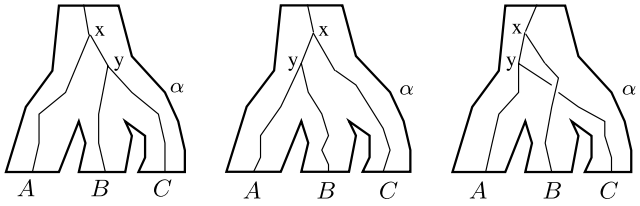

FIG. 3. Gene trees evolving within the same species polytomy can have different binary topologies.

equally likely (Pamilo and Nei, 1988). Figure 3 shows all three binary branching patterns that are possible in a polytomy with three children.

In our model, disagreement between a binary gene tree and a non-binary species tree is evidence that divergence in the gene tree resulted from gene duplication or incomplete lineage sorting. When the species tree is non-binary, three cases must be considered. Where there is no incongruence, gene tree divergence is attributed to speciation. Second, some divergences can only be explained by gene duplication. Obviously, a duplication must have occurred in any gene family that has two or more members in the same species. But even when no contemporary species contains more than one family member, there are cases where topological disagreement can only be explained by a duplication. Third, in some cases, it is not possible to determine whether the disagreement is due to incomplete lineage sorting or duplication (Slowinski and Page, 1999).

These issues are illustrated in Figure 4. Disagreement between the gene tree in Figure 4b and the species tree in Figure 4a can be explained by two scenarios, depicted in Figures 4c and d. In Figure 4c, the discordance at x is explained by a duplication, followed by losses in B and β. In Figure 4d, the divergence at x is explained by allelic variation in the ancestral population at α, followed by retention of different alleles in different lineages in T_S. Based on the information available in Figure 4, it is impossible to determine with certainty whether node x represents incomplete lineage sorting or gene duplication. In contrast, the incongruence at node y between the species tree in Figure 4a and the gene tree in Figure 4b can only be explained by a duplication. This can be seen in Figure 4d, which shows the gene tree embedded in the species tree. Two copies of the gene are present in the lineage from α to β, indicating that a duplication must have occurred at their most recent common ancestor, y.

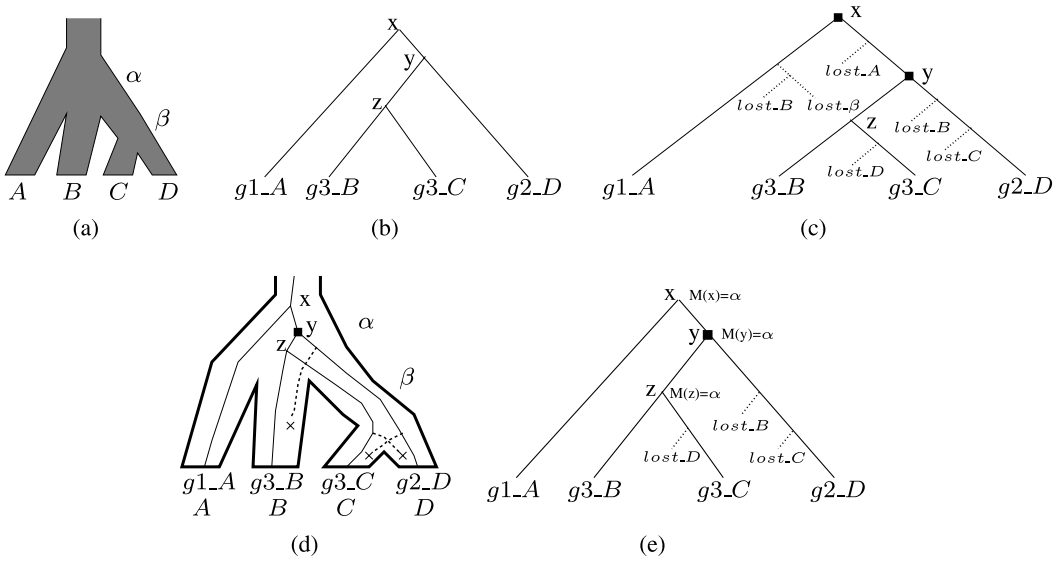

FIG. 4. **(a)** A species tree with a polytomy at α. **(b)** A hypothetical gene tree sampled from the species in (a). **(c)** The gene tree from (b), which has been reconciled using the LCA algorithm. **(d)** The hypothetical gene tree embedded in species tree. **(e)** The gene tree labeled with required duplications and losses. Black squares indicate (required) duplications. Losses are represented by dotted lines.

Since disagreements in the branching pattern of a binary gene tree and a non-binary species tree may be evidence of a duplication or incomplete lineage sorting, we need a formal basis to distinguish between *required* duplications—those disagreements that can only be explained by a duplication—and *conditional* duplications—those disagreements that can be explained by either a duplication or a deep coalescence event. LCA reconciliation can recognize nodes that did not arise through speciation but cannot distinguish between required and conditional duplications. In Figure 4, all three internal nodes map to species α. According to Equation (2), since $M(x) = M(y) = M(z) = \alpha$, duplications are inferred at both x and y. Thus, LCA reconciliation recognizes disagreement at x and y, but cannot determine that x is a conditional duplication and y a required duplication. An additional test for distinguishing required and conditional duplications is needed.

Recall that all binary trees with k leaves are equally compatible with a species polytomy with k children. Therefore, we can treat a polytomy s as a set of hypotheses, or *binary resolutions of* s. For each polytomy $s \in V_S$, let $H(s)$ be the set of all possible binary trees, rooted at s, whose leaves are the children of s. Formally, given the k-tomy $s \in V_S$, let $H(s) = \{T_i | L(T_i) = C(s)\}$, where T_i is a binary tree such that the leaves of T_i are the children of s. For example, if node s is the trichotomy α in Figure 4a, then $H(\alpha) = \{(A, (B, \beta)), (B, (A, \beta)), (\beta, (A, B))\}$. In addition, let $H^*(T_S)$ be the set of all possible binary trees obtained by replacing each polytomy $s_j \in V_S$ with each tree $T_{ij} \in H(s_j)$. In other words, $H^*(T_S)$ is the set of all possible binary resolutions of T_S. For Figure 4a, $H^*(T_S) = \{(A, (B, (C, D))), (B, (A, (C, D))), ((C, D), (A, B))\}$. Note that for a given $T' \in H^*(T_S)$, every node $s \in T_S$ corresponds to a node in T'; however, T' will also contain nodes that do not correspond to any node in T_S. The cardinality of $H^*(T_S)$ is $\prod_{\forall s_j \in T_S} |H(s_j)|$, which is equivalent to $\prod_j n_j$, where $n_j = \frac{(2k_j - 3)!}{2^{k_j - 2}(k_j - 2)!}$ and $k_j = |C(s_j)|$ (Li, 1997). If T_S is binary, then $H^*(T_S) = \{T_S\}$.

We now use $H^*(T_S)$ to characterize formally the properties of the gene and species tree that determine when a duplication is required. When reconciling T_G with every $T' \in H^*(s)$, if $g \in V_G$ is a duplication in every reconciliation, then a duplication *must* have occurred at g. If at least one, but not all reconciliations indicate a duplication at g, then a deep coalescence event *may* have occurred. Formally:

Definition 3.1. *$\forall\ T' \in H^*(T_S)$, reconcile T_G with T'. Given $g \in V_G \setminus root(T_G)$,*

- *$p(g)$ is a* required duplication *if $\forall\ T' \in H^*(T_S)$, $M(g) = M(p(g))$.*
- *$p(g)$ is a* conditional duplication *if $\exists\ T' \in H^*(T_S)$ s.t. $M(g) = M(p(g))$ and $p(g)$ is not a required duplication.*

Notice that for the trees in Figure 4, every T' would infer a duplication node at node y; however, this is not the case for node x. For these trees, $M(y) = M(z)$ for every $T' \in H^*(T_S)$. Therefore, under Definition 3.1, node y is a required duplication. On the other hand, node x is a conditional duplication since there is a binary resolution in $H^*(T_S)$, namely $(A, (B, (C, D)))$, that does not infer a duplication. Recall, however, that under LCA reconciliation, a duplication event is inferred at both nodes x and y, since $M(x) = M(y) = M(z)$.

4. IDENTIFYING DUPLICATIONS

Reconciliation with a non-binary species tree requires determining whether a given node is a required duplication, a conditional duplication, or a speciation. Formally, the problem of reconciliation with non-binary species trees is defined as follows:

Duplication Inference for Non-Binary Species Trees

Input: A rooted, arbitrary species tree, T_S, a rooted, binary gene tree, T_G, and a mapping from $L(T_S)$ to $L(T_G)$.

Output: The minimum number of duplications required to reconcile of T_G and T_S. The most parsimonious duplication histories, where a duplication history is a reconciled gene tree in which every node is designated as a required duplication, a conditional duplication, or a speciation.

Definition 3.1 provides a formal basis for classifying nodes, but cannot be the basis of an efficient algorithm, since $H^*(T_S)$ grows superexponentially with the size of polytomies in T_S. LCA reconciliation is not a suitable solution, since it identifies both conditional and required duplications, but cannot distinguish between them.

To see why this is true, recall that the presence of descendants of both children of g in the same lineage of T_S is evidence of duplication at g. To infer a duplication at g in the binary case, it is sufficient to determine that g and a child of g were both present in $M(g)$, because (in the absence of loss) complete lineage sorting implies that both lineages descending from g must be inherited by both species lineages descending from $M(g)$. However, when incomplete lineage sorting is possible, this is no longer true. The coexistence of g and one of its children in the same ancestral species (i.e., Equation (2)) is not a sufficient condition to determine whether they coexisted in the same lineage. For example, in Figure 4d, the left child of x is only present in the lineage leading to A and the right child of x is only present in the lineages leading to B and β. No subtree descending from a child of $M(x)$ contains descendants of both $l(x)$ and $r(x)$, indicating that x is not a duplication node. In contrast, y is a required duplication since both of its children are present on the edge from α to β. Equation (2) would correctly recognize the required duplication at y, but incorrectly infer a required duplication at x as well, since $M(x) = M(r(x))$.

As the above example shows, Equation (2) is not sufficient to infer required duplications when the species tree is non-binary. In particular, the mapping $M(\cdot)$ is not sufficiently informative to distinguish between conditional duplications and required duplications. In order to infer required duplications, information is needed about the descendants of $M(g)$ in which the descendants of g were present.

Here we propose a new mapping and show how it can be used to make this distinction. We construct a mapping from each node g in T_G to a set of species in T_S. A straightforward approach would be to label each node, g, in the gene tree with all nodes (both leaves and internal nodes) in the species tree in which the gene was present. Using this mapping, a required duplication is inferred at g if the intersection of the sets of its children is non-empty. The size of the sets labeling the nodes in the gene tree grows with the height of the tree and can contain as many as $O(|V_S|)$ elements. However, it is sufficient to store only the children of $M(p(g))$ in which descendants of g must have been present, as follows:

Definition 4.1. *Define* $\hat{N} : V_G \setminus root(T_G) \to V_S^+$ *to be*

$$\hat{N}(g) = \begin{cases} \{M(g)\}, & \text{if } M(p(g)) \in L(T_S) \\ \{h | h \in C(M(p(g))) \wedge \exists\, v \in L(g) \ni h \geq_S M(v)\}, & \text{otherwise.} \end{cases}$$

where V_S^+ *is the powerset of* V_S*, excluding the empty set.*

Note that $\hat{N}$ is defined on every node in T_G except the root. The size of this mapping at any given node is bounded by the size of the largest polytomy in T_S, yet this mapping is sufficiently informative to identify required and conditional duplications, as we prove in Theorem 4.1.

Theorem 4.1. *A node* g *is a required duplication iff* $\hat{N}(r(g)) \cap \hat{N}(l(g)) \neq \emptyset$.

Proof. There exists at least one x such that $x \in \hat{N}(l(g)) \cap \hat{N}(r(g))$. Therefore, for all $T' \in H^*(T_S)$, $x \leq_S M(l(g))$ and $x \leq_S M(r(g))$. This requires that either $M(l(g)) = M(r(g))$ or one is a descendant of the other. Thus, g meets the duplication criterion for binary gene and binary species trees for every $T' \in H^*(T_S)$, and therefore is a required duplication.

It is necessary to show that whenever $\hat{N}(r(g)) \cap \hat{N}(l(g)) = \emptyset$, there exists at least one element of $H^*(T_S)$ that does not imply a duplication at g. Any $T' \in H^*(T_S)$ that has all members of $\hat{N}(l(g))$ in the left subtree of $M(g)$ and all members of $\hat{N}(r(g))$ in the right subtree of $M(g)$ will meet this criterion. At least one such tree must exist since $\hat{N}(r(g))$ and $\hat{N}(l(g))$ do not intersect. ■

Figure 5 shows the mapping $\hat{N}(\cdot)$ for the gene tree in Figure 4d. This mapping correctly infers a required duplication at y since $\hat{N}(l(y)) \cap \hat{N}(r(y)) = \{B, \beta\} \cap \{\beta\} = \{\beta\} \neq \emptyset$. It also correctly identifies a conditional duplication at node x, since $\hat{N}(l(x)) \cap \hat{N}(r(x)) = \{A\} \cap \{B, \beta\} = \emptyset$.

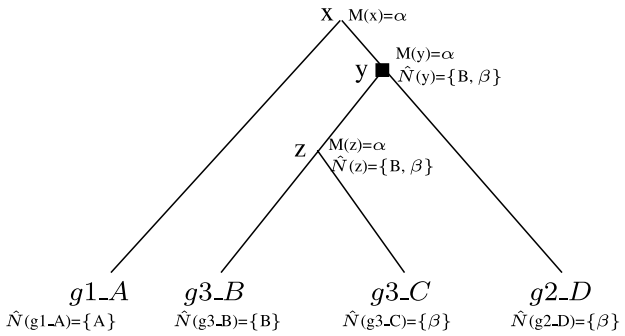

FIG. 5. The gene tree from Figure 4e labeled with the $\hat{N}(\cdot)$ mapping. Losses are not represented in this figure.

The algorithm for inferring required and conditional duplications using $\hat{N}(\cdot)$ is given in Algorithm 1. $\hat{N}(g)$ is calculated with a postorder traversal of T_G. To ensure that the set $\hat{N}(\cdot)$ is composed only of children of $M(p(g))$, Algorithm 1 executes a climbing step that replaces the set of nodes in $\hat{N}(g)$ with the child of $M(p(g))$ which is ancestral to them. The *climb* procedure prevents $|\hat{N}|$ from growing larger than k_S. Algorithm 1 infers a required duplication at g if the intersection of the sets $\hat{N}(l(g))$ and $\hat{N}(r(g))$ is non-empty. A conditional duplication is inferred if the node is not a required duplication, but is a duplication under LCA reconciliation.

Algorithm 1.

```
reconcile( g )
 1  if ( g.isLeaf() )
 2     M(g) = species of g
 3     N̂(g) = {M(g)}
 4     return
 5  // INTERNAL NODE CASE
 6  // descend first
 7  reconcile(l(g)); reconcile(r(g))
 8  M(g) = LCA(M(l(g)), M(r(g)))
 9  // Update N̂(·) for children by climbing
10  N̂(l(g)) = climb(l(g), g); N̂(r(g)) = climb(r(g), g)
11  N̂(g) = N̂(l(g)) ∪ N̂(r(g))
12  if ( N̂(l(g)) ∩ N̂(r(g)) ≠ ∅ )
13     g is Required Duplication
14  else if ( M(g) == M(l(g)) or M(g) == M(r(g)) )
15        g is Conditional Duplication

climb( c, g )
16  select x from N̂(c) at random
17  if ( x == M(g) or p(x) == M(g) )
18     return x
19  while ( p(x) ≠ M(g) )
20     // climb
21     x = p(x)
22  return {x}
```

As shown in Theorem 4.2, Algorithm 1 produces the same results as LCA reconciliation when the species tree is binary.

Theorem 4.2 (Equivalence with LCA Reconciliation for Binary Species Trees). *Let T_G be a binary gene tree reconciled with a binary species tree. $\hat{N}(r(g)) \cap \hat{N}(l(g)) \neq \emptyset$ iff $M(r(g)) = M(g)$ and/or $M(l(g)) = M(g)$.*

Proof. This follows directly from Theorem 4.1.

Suppose that there exists $h \in C(g)$, such that $M(h) = M(g)$, but $\hat{N}(r(g)) \cap \hat{N}(l(g)) = \emptyset$. Either $\hat{N}(l(g))$ and $\hat{N}(r(g))$ are both equal to $\{M(g)\}$ or $\hat{N}(l(g))$ and $\hat{N}(r(g))$ both contain children of $M(g)$. In the former case, $\hat{N}(l(g))$ and $\hat{N}(r(g))$ are not disjoint, leading to a contradiction. In the latter case, since $M(g)$ has only two children and the sets are disjoint, one set must contain the right child and the other the left child of $M(g)$. Thus, there is no $h \in C(g)$, such that $M(h) = M(g)$, leading to a contradiction. ■

The computational complexity of this algorithm is considered in Section 5.1, where we present an algorithm that infers losses in addition to conditional and required duplications in a single traversal of the tree.

5. INFERRING GENE LOSSES

The goal of loss inference under the parsimony criterion is to identify the loss history that requires the minimum number of losses needed to explain the data. A loss history is a set of gene losses, in which each loss is assigned to an edge in the gene tree and a node in the species tree. For a given binary gene tree and binary species tree, there is exactly one most parsimonious loss history. Each loss can be unambiguously assigned to exactly one edge in T_G, and associated with one node in T_S. Moreover, it is possible to determine the set of losses assigned to an edge $(g, p(g))$ by comparing $M(g)$ and $M(p(g))$, without considering losses on any other edge of the gene tree. The total number of losses in the most parsimonious history can be determined by inferring losses on each edge independently and summing over all edges.

In contrast, when T_S is non-binary, a reconciliation may have more than one equally parsimonious loss history. Under specific circumstances, described in detail in the next section, a species polytomy will result in ambiguous losses, losses that may be assigned to one of several edges in the gene tree. The reconciliation does not provide enough information to fully resolve the temporal order of these losses relative to other events in the same gene tree lineage.

The problem of ambiguous losses is illustrated by the following example: Figure 6a shows a species tree, T_S, with a single polytomy, β, and a gene tree, T_G, representing the evolution of a family of genes, g, drawn from the species in $L(T_S)$. There is no member of this gene family in species B, indicating the loss of a gene in B. Comparison of T_G and T_S identifies three edges in T_G from which this gene (denoted *lost_B*) could have diverged, but is not sufficient to determine which edge is preferred. The divergence of *lost_B* may have occurred before the separation of *g_C* from the ancestor of *g_D* and *g_E* (Fig. 6b) or after that separation (Fig. 6c). Alternatively, *lost_B* may be most closely related to *g_C* (Fig. 6d).

In gene families in which two or more losses occurred, interactions between losses that can be assigned to the same edge of the gene tree must be considered. Although it is not possible to determine exactly when an ambiguous loss occurred, it is possible to identify the set of *permissible edges* for a given loss. The set of permissible edges is a (not necessarily proper) subset of a set of contiguous edges, called a Polytomy Connected Component (PCC), which is defined formally in Section 5.2. Each ambiguous loss is associated with exactly one PCC. A gene tree may have several PCCs; these are always disjoint. Interactions between ambiguous losses assigned to the same PCC influence the total number of losses. Two factors contribute to the interactions between losses: Losses that occurred in sibling species and that are assigned to the same edge in T_G may be replaced by a single loss in a common ancestor, decreasing the total loss count. In addition, interaction of ambiguous losses with duplications in the gene tree affects the total number of losses inferred.

The first factor arises because a single loss in an ancestral species is more parsimonious than simultaneous, independent losses in the contemporary species descended from that ancestor. In an algorithmic context, we refer to the process of inferring such ancestral losses as *combining losses*. In the binary case, a single ancestral loss may be inferred if the same member of a gene family is absent from all leaves

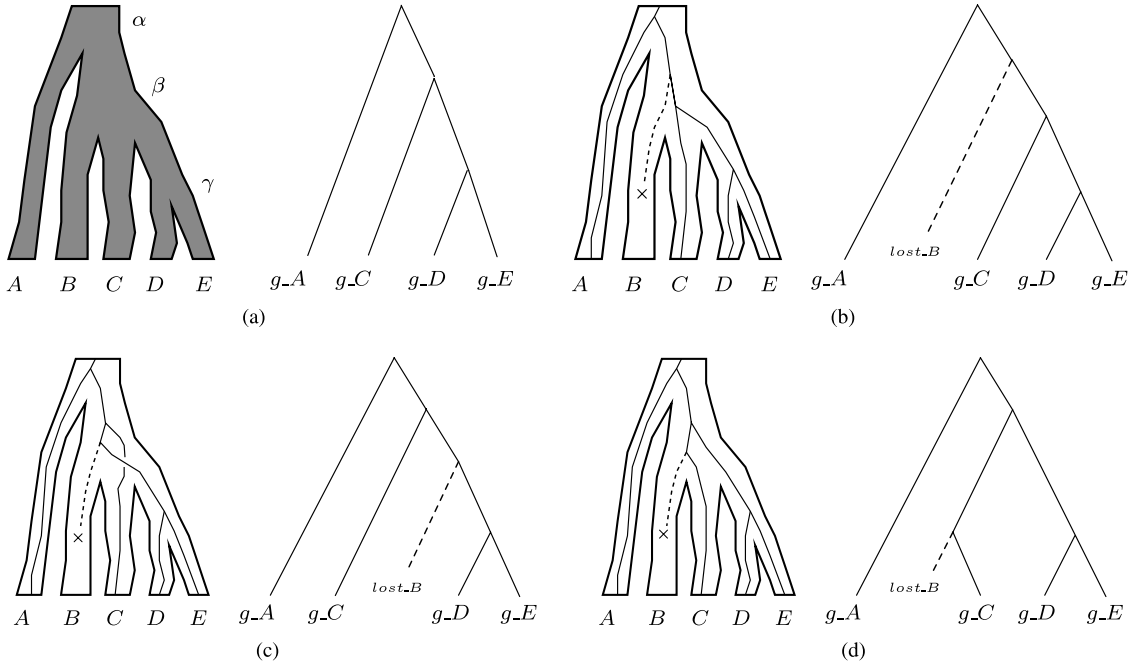

FIG. 6. Loss ambiguity. **(a)** Hypothetical species and gene tree. T_S has a polytomy at β and a gene has been lost in species B in T_G. **(b)**, **(c)**, and **(d)** The different hypotheses of when the lost occurred are shown as both an embedded tree and a separate gene tree.

of some subtree of T_S. When the species tree is non-binary, losses that correspond to leaves of subtrees whose roots are the children of a polytomy may be combined. Formally, given a set, $\mathcal{L}$, of losses assigned to the same edge, if there is a one-to-one mapping between $\mathcal{L}$ and the leaves of a subtree t in some $T' \in H^*(T_S)$, a single loss in $root(t)$ can be inferred. It is not necessary to enumerate all trees in $H^*(T_S)$ to determine whether a given set of losses corresponds to a single, ancestral loss T_S. If there exists a set of subtrees in T_S such that there is a one-to-one mapping between $\mathcal{L}$ and the leaves of the subtrees, and the roots of the subtrees are all children of the same polytomy, then the losses in $\mathcal{L}$ may be combined. Note that a reconciled gene tree may include losses in ancestral species that are not found in T_S. In this case, the loss is labeled with the set of children of the polytomy that are descendants of $root(t)$ in T'.

The particular edge within the permissible set to which a loss is assigned determines whether or not it can be combined with other losses and, hence, the total number of losses inferred. For example, the gene family in Figure 7c is not represented in species B. The set of permissible edges for *lost_B* is shown in bold. A second loss occurred in species D. The set of permissible edges for *lost_D* is not shown, but it overlaps with the set of edges for *lost_B*. In particular, both losses can be assigned to the edge (w, x). In that case, a single ancestral loss can be inferred in their common ancestor, as seen in Figure 7d. This is permissible because there exists a tree in $H^*(T_S)$ in which B, D, and their common ancestor form a monophyletic subtree. Notice that, although *lost_B* and *lost_F* can both be assigned to the edge $(x, g4\_E)$, they do not correspond to the leaves of any subtree in $H^*(T_S)$, and cannot be combined.

The second factor influencing total loss count is loss duplication. If the set of permissible edges for a given ambiguous loss contains a duplication node, the position of the loss with respect to this duplication will influence the total number of losses. Assignment of the loss to an edge below the duplication implies that the gene was duplicated before the loss occurred, requiring that two losses be inferred, one in each subtree below the duplication. If the loss is assigned on an edge above the duplication, only a single loss is inferred. While assigning losses to edges above duplications usually results in fewer losses, the number of losses will decrease when a loss is assigned below a duplication if both copies can then be combined with other losses. For example, in Figure 7c, *lost_B* may be assigned to edges both above and below the duplication at w. If *lost_B* is assigned to edges below w, two losses in B are inferred, one on edge

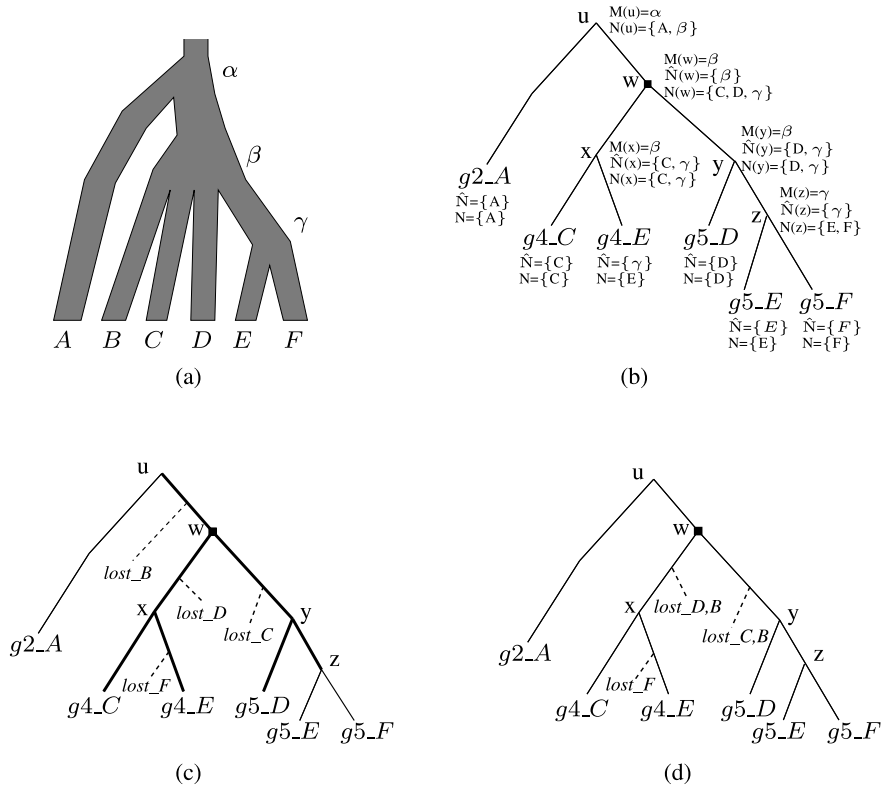

FIG. 7. **(a)** A species tree with a polytomy, β. **(b)** A hypothetical gene tree that has been reconciled with the species tree in (a). This gene tree is annotated with the mappings $M(\cdot)$, $\hat{N}(\cdot)$ and $N(\cdot)$. Losses are not represented in this tree. **(c)** The gene tree (b) annotated with the loss placement inferred by Algorithm 2. **(d)** The gene tree (b) annotated with one possible optimal placement of polytomy losses inferred by Algorithm 3.

$(w, l(w)) = (w, x)$ and one on $(w, r(w) = (w, y))$. As seen in Figure 7d, this results in two ancestral losses: *lost_D*, B and *lost_C*, B.

Due to the interactions between multiple ambiguous losses and duplications in the same PCC, and in contrast to reconciliation with binary species trees, the total number of losses in the most parsimonious history cannot be determined by considering each edge independently. Inferring the minimum number of losses requires obtaining the set of lowest cost assignments from the set of all possible edge assignments for each PCC. The problem of inferring the set of most parsimonious loss histories for a given binary gene tree and binary or non-binary species tree is stated formally as follows:

Non-Binary Loss Inference

Input: A rooted, arbitrary species tree, T_S, a rooted, binary gene tree, T_G, and a mapping from $L(T_G)$ to $L(T_S)$.

Output: The minimum number of losses required to reconcile T_G and T_S. The set of all most parsimonious loss histories, where a loss history is an assignment of each loss to an edge in T_G and a species node in some tree in $H^*(T_S)$.

Given these considerations, we present two algorithms for inferring losses, described in Sections 5.1 and 5.2. For a given gene tree, T_G, and species tree, T_S, we report the total number of losses in T_G and assign individual losses to edges in T_G. One is an exact method, which considers all possible assignments of losses to edges in T_G and selects the assignment that minimizes the number of inferred losses. This method runs in $O(|V_G|k_S 2^{2k_S})$, and can return each additional optimal loss history in $O(|V_S|)$. The second strategy is a heuristic, and returns a single loss history. The heuristic runs in $O(|V_G| \cdot (k_S + h_S))$, and although not guaranteed to return an optimal history, does very well in practice, as described in Section 7.

The heuristic is a simpler procedure, and is used as a framework for the exact algorithm, so we present it first.

5.1. Heuristic for inferring loss histories

The heuristic uses a greedy strategy that makes loss assignment decisions at each edge, without considering interactions with losses inferred on other edges. The strategy is to minimize duplicated losses by assigning each ambiguous loss to the permissible edge closest to the root. This guarantees that the loss will not be unnecessarily assigned below a duplication node, leading to the inference of two losses, instead of one. We do not attempt to optimize combined losses; however, after all losses are assigned, losses that satisfy the appropriate criteria are combined. This strategy will occasionally return a suboptimal loss history because it will fail to identify situations where assigning a loss below a duplication will allow the resulting copies to be combined with other losses, thus decreasing rather than increasing the total loss count. However, in practice, this happens rarely. For example, the heuristic will fail to find the optimal loss history for the gene tree in Figure 7b. Instead, it will return the reconciliation shown in Figure 7c, where *lost_B* is assigned to (u, w). If *lost_B* were assigned below the duplication to (w, x) and (w, y), a lower cost tree could be obtained, shown in in Figure 7d.

The heuristic traverses the tree in post order. At each edge, it (1) determines whether one or more losses should be assigned to that edge and (2) identifies the species in which the losses occurred. The first step is achieved by the application of three tests, described below. The greedy strategy, which assigns ambiguous losses as close to the root as permissible, is a natural outcome of the post order traversal. The permissible edge closest to the root is reached first, allowing the heuristic to assign the loss to the desired edge without explicitly determining the set of permissible edges for each loss.

The species in which the losses occurred are inferred in the second step, by comparing sets of species nodes assigned to each node $g \in V_G$. This is similar to duplication inference for non-binary species trees, but in the case of losses, the set comparisons are more complex. Three different sets of nodes are considered for every edge $e = (g, p(g))$:

- $C(M(g))$: The set of all children of $M(g)$.
- $\hat{N}(g)$: The set of children of $M(p(g))$ that contain a descendant of g
- $N(g)$: The set of children of $M(g)$ that contain a descendant of g.

The first two sets we have encountered in the previous section. The third, N, is defined as follows:

Definition 5.1. *Define* $N : V_G \to V_S^+$ *to be*

$$N(g) = \begin{cases} \{M(g)\}, & \text{if } M(g) \in L(T_S) \\ \{h | (h \in C(M(g)) \wedge \exists\ v \in L(g) \ni h \geq_S M(v))\}, & \text{otherwise.} \end{cases}$$

Note that unlike $\hat{N}(g)$, $N(g)$ is defined for $root(T_G)$. See Figure 7b for an example of $\hat{N}(g)$ and $N(g)$.

Each of the following tests corresponds to one of the three situations that can incur a loss. The heuristic applies these tests to each edge, $e = (g, p(g))$, in E_G and assigns the inferred losses to e. Note that these tests are also used in the exact algorithm described in Section 5.2, although in the exact algorithm a further optimization step is applied following the initial identification of losses.

Test 1: If $M(g) \neq M(p(g))$ and $p(M(g)) \neq M(p(g))$, traverse the path from $M(g)$ to $M(p(g))$ in T_S, inferring a loss diverging from each intermediate species along this path (lines 29–31 in Algorithm 2).

This test is applied to all edges in T_G, whether associated with a binary node or polytomy in T_S. If a node and its parent map to different nodes in T_S, we expect those nodes to correspond to child and parent nodes in the species tree. Otherwise, genes in the intervening species must have been lost. The procedure to infer these skipped losses is carried out in the climb procedure and is analogous to that used in LCA reconciliation, described in Section 2. An example of this test is given in Figure 7c. The test is invoked on edge $(x, g4\_E)$, because $M(x) \neq M(g4\_E)$ and $M(x) = \beta$ is not the parent of species E. The climb procedure will recognize that a gene is missing between E and β and will infer a loss in species F.

Test 2: If $p(g)$ is a required duplication, then losses are inferred in the species in $N(p(g)) \setminus \hat{N}(g)$ at e (lines 18–21 in Algorithm 2).

This test is applied to all edges where $p(g)$ is a required duplication, whether associated with a binary node or polytomy in T_S. Note that if $M(p(g))$ is binary and $M(p(g)) = M(g)$, then $N(p(g)) = \hat{N}(g)$ and no losses are inferred. Thus, this test reduces to that used in LCA reconciliation for binary nodes in T_S. When $M(p(g))$ is polytomy, losses may occur even when $M(p(g)) = M(g)$, in contrast to binary reconciliation. Both the edges (w, x) and (w, y) in Figure 7c meet the criteria in this test. For example, w is a required duplication, and comparing $N(w)$ with $\hat{N}(x)$ indicates that a loss occurred in D.

Test 3: If $M(g)$ is a polytomy and $M(p(g)) \neq M(g)$, then losses are inferred in the species in $C(M(g)) \setminus N(g)$ at e (lines 22–24 in Algorithm 2).

This test is only applied when $p(g)$ is a speciation node and $M(g)$ is a polytomy. It verifies that each child of $M(g)$ contains a descendant of g. If not, one or more losses must be inferred. This test is applied to (u, w) in Figure 7c, since $M(w)$ is a polytomy and $M(w) \neq M(u)$. A loss is inferred in $C(M(w)) \setminus N(w) = \{B, C, D, \gamma\} \setminus \{C, D, \gamma\} = B$. Note that although this loss could be assigned to any of the dotted edges in Figure 8, the greedy strategy assigns the loss to edge (u, w) the first time it is encountered.

The heuristic procedure to infer losses, described in Algorithm 2, calculates $N(\cdot)$ and $\hat{N}(\cdot)$ and applies each of the three tests in a single postorder traversal of T_G. There is only one optimal solution under the assumptions of this heuristic. All losses can be inferred in a single pass. Duplications are also inferred during this postorder traversal, as described in Section 4.

Algorithm 2.

```
reconcile( g )
 1  if ( g.isLeaf() )
 2     M(g) = species of g
 3     N(g) = {M(g)}
 4     return
 5  // INTERNAL NODE CASE
 6  // descend first
 7  reconcile(l(g)); reconcile(r(g))
 8  M(g) = LCA(M(l(g)), M(r(g)))
 9  calculateRequiredDuplication( g )
10  if ( g ≠ Required Duplication )
11     if ( M(g) == M(l(g)) || M(g) == M(r(g)) )
12        g is Conditional Duplication

calculateRequiredDuplication( g )
13  N̂(l(g)) = climb(l(g), g)
14  N̂(r(g)) = climb(r(g), g)
15  N(g) = N̂(l(g)) ∪ N̂(r(g))
16  if ( N̂(l(g)) ∩ N̂(r(g)) ≠ ∅ )
17     g is Required Duplication
18     // duplication losses for left child
19     Losses(l(g)) += N(g) \ N̂(l(g))
20     // duplication losses for right child
21     Losses(r(g)) += N(g) \ N̂(r(g))
```

(*continued*)

Algorithm 2. *(Continued)*

```
climb( c, g )
22  // general polytomy losses
23  if ( M(c) ∉ L(T_S) && M(c) ≠ M(g) )
24    Losses(c) += C(M(c)) \ N(c)
25  select x from N(c) at random
26  if ( x == M(g) || p(x) == M(g) )
27    return N(c)
28  while ( p(x) ≠ M(g) )
29    // skipped losses
30    if ( p(x) ≠ M(c) )
31      Losses(c) += Siblings(x)
32    // climb
33    x = p(x)
34  return {x}
```

Theorem 5.1. *Algorithm 2 computes required and conditional duplications, as well as heuristic losses, in $O(|V_G| \cdot (k_S + h_S))$, where k_S is the outdegree of the largest polytomy in T_S, and h_S is the height of T_S.*

Proof. At every internal node $g \in V_G$, $N(g)$ is initialized with $\hat{N}(l(g)) \cup \hat{N}(r(g))$. $|\hat{N}(\cdot)|$ is bounded by k_S. Using a suitable data structure, this step can be achieved in $O(\log(k_S))$ time per node. The climb routine is applied to every node in T_G. For any given path from $\lambda \in L(T_G)$ to $\rho = root(T_G)$, we will climb in total from $M(\lambda)$ to $M(\rho)$. Thus, the total cost of calls to climb is $O(|V_G| \cdot h_S)$.

Using fast Least Common Ancestor queries, $M(\cdot)$ can be calculated in $O(|V_G|)$ time for the entire tree (Bender and Farach-Colton, 2000). Once $M(\cdot)$ has been calculated, testing for conditional duplications takes constant time per node. Testing for required duplication requires calculation of the intersection of $\hat{N}(l(g))$ and $\hat{N}(r(g))$. This operation takes $O(k_S)$ per node. Combining these costs, the total running time is $O(|V_G| \cdot (k_S + h_S))$. ■

5.2. Inferring optimal loss histories

In this section, we present an algorithm that considers all possible loss assignments to obtain the minimum number of combined losses. Unlike the greedy heuristic in Algorithm 2, this algorithm finds all

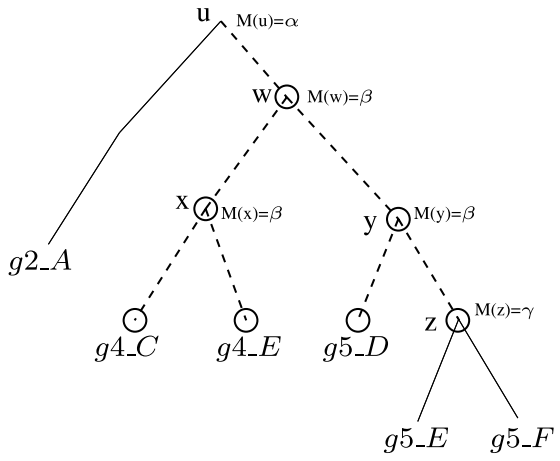

FIG. 8. The Polytomy Connected Component in Figure 7.

optimal assignments, but with increased computational complexity: the running time is exponential in k_S. However, the implementation is fast in practice because k_S is typically small (see Section 6). Moreover, the performance is enhanced by memoization, as explained below.

Before describing the algorithm, we introduce the basic approach to minimizing combined losses using the example in Figure 7. Three losses in Figure 7c, *lost_B*, *lost_C* and *lost_D*, are ambiguous losses that can be assigned to more than one edge in T_G. Recall that *lost_B* can be moved below the duplication at w, resulting in two copies of *lost_B*, one in each child subtree of w. While this would normally increase the number of losses, one copy of *lost_B* can be combined with *lost_C*, while the other can be combined with *lost_D*. This new assignment reduces the number of inferred losses from four to three, as shown in Figure 7d.

A PCC is a set of contiguous edges in T_G, with the property that all internal nodes map to the same polytomy in T_S. Each ambiguous loss is associated with exactly one PCC and can be assigned to a (not necessarily proper) subset of its edges. In particular, if a loss can be assigned to some edge e, it can also be assigned to any edge below e in the PCC. Formally, we define a PCC as follows:

Definition 5.2. *A node $r_c \in V_G$ is the component root of a distinct PCC if $M(r_c)$ is a polytomy and $(r_c = root(T_G) \vee M(r_c) \neq M(p(r_c)))$. Let X be the set of nodes that contains r_c and all $g \in V_G$ such that $g <_G r_c$ and $M(p(g)) = M(r_c)$. Y is the set of edges $\{(x, (p(x))|x \in X \setminus root(T_G)\}$. X and Y are the sets of nodes and edges in this PCC, respectively.*

The gene tree in Figure7b has a single PCC, shown in Figure 8. Edges in the PCC are drawn with dotted lines. The nodes in the PCC are circled. The component root of this PCC, which corresponds to the polytomy β in the species tree in Figure 7a, is w. Note that for every node $v \in X$, all nodes on the path from v to the r_c map to the associated polytomy. In our example, nodes $M(w) = M(x) = M(y) = \beta$.

The entire reconciliation procedure involves two passes through the tree. In the first pass, a modified version of Algorithm 2 (not shown) traverses the tree in postorder, calculating $\hat{N}(\cdot)$ and $N(\cdot)$, inferring duplications, and inferring losses associated with binary nodes by invoking tests 1 and 2, described in Section 5.1. Test 1 is applied to any edge $e = (g, p(g))$ where $p(g)$ is a required duplication and $M(p(g))$ is binary. Test 2 is applied to every edge in T_G. In the second pass, Algorithm 3 infers optimal assignments of ambiguous losses for each PCC, using a dynamic programming strategy. Algorithm 3 visits each node $g \in V_G$ and determines whether g is the component root of a new PCC (line 3). If so, it calls *ProcessComponent*, which traverses the PCC in postorder to find optimal assignments of ambiguous losses to edges in that PCC. For each node, g, in the PCC, *ProcessComponent* calculates the minimum cost of any assignment of losses to the edge $(p(g), g)$ or to edges in the subtree rooted at g. Tables Γ and Υ store the cost and assignment information for g, respectively. In addition, the total number of optimal loss histories in the subtree rooted at g is determined and stored in the variable $\mathcal{H}$. After these cost tables have been calculated, *AddCombinedLosses* uses a preorder traversal of the PCC to generate an optimal loss assignment from the values stored in Γ and Υ.

The core of the algorithm is the calculation of Γ and Υ for each internal node g. *ProcessComponent* considers all possible ways of assigning losses from the set $\mathcal{L} = C(M(r_c)) \setminus \tilde{N}(g)$ to g, where r_c is the root of the current component and $\tilde{N}(g)$ is $N(g)$ if $g = r_c$ but is $\hat{N}(g)$ otherwise. It calls *CalculateCost(g, f_{in})* for each element, f_{in}, of the power set of $\mathcal{L}$. The parameter f_{in} is a set of losses, represented as a set of species, that must be be assigned at or below g. *CalculateCost* considers two cases: a special case that is invoked if g is a required duplication, which handles loss duplication, and a second case if g is a conditional duplication or a speciation. In both cases, the cost of f_{in} at g is the minimum cost of assigning a subset of losses in f_{in} to the edge above g, plus the cost of losing the rest of f_{in} below g. The set of species $f_{\text{out}} \subseteq f_{\text{in}}$ represents losses that occur below g; the set of remaining species, $f_{\text{in}} \setminus f_{\text{out}}$, are assigned to $(p(g), g)$. For a given g and f_{in}, *CalculateCost* determines the optimal assignment over all possible subsets f_{out}. The cost is calculated for each value of f_{out} and f_{in}, and the lowest cost for each g and f_{in} is stored. Note that it is not necessary to explicitly test each possible value of f_{out} for a given g and f_{in}. By enumerating the smallest values of f_{in} first at line 21, we can store intermediate minimum costs and assignments in Γ_{mem} and Υ_{mem}. By using these memoized values, we only have to explicitly test $\binom{|f_{\text{in}}|}{|f_{\text{in}}|-1} = |f_{\text{in}}| - 1$ values of f_{out} (on lines 31 and 32, and 65 and 66) for each value of g and f_{in}.

CalculateCost also stores the assignment of losses to edges associated with that cost. $\Upsilon(g, f_{\text{in}})$ records, in a tuple, the species which must be lost in the subtrees rooted at the left and right children of g. Each optimal loss assignment is stored in a list in Υ. These values are used in *AddCombinedLosses* to select an optimal loss placement. After calculating Γ and Υ, an optimal loss assignment is selected using a preorder traversal of the PCC. Although the pseudocode in Algorithm 3 only selects a single, arbitrary placement, all possible placements can be generated by selecting each permutation of optimal placements from Υ for all nodes in the PCC. This part of the algorithm is mostly bookkeeping and is omitted for brevity.

The running time to return one solution is $O(|V_G|k_S 2^{2k_S})$. The exponential term is due to the enumeration of the power set of $\mathcal{L}$ in *ProcessComponent* followed by the nested enumeration of an additional power set in *CalculateCost*. This enumeration is carried out potentially for each vertex in V_G and the size of each of the two power sets is $O(2^{k_S})$. For each node we also iterate over the elements of the set f_{in} to compute each f_{out}. The size of these sets is bounded by $O(k_S)$. Thus, the total cost of calculating the tables Γ and Υ is $O(|V_G|k_S 2^{2k_S})$. Each additional optimal loss history can be generated in $O(|V_S|)$. The running time to calculate Γ and Υ is improved by memoization in *CalculateCost* by an order of $2^{k_S}/k_S$, from $O(|V_G|2^{3k_S})$.

Algorithm 3.

```
CombinedLosses( g )
 1  if ( g.isLeaf() )
 2    return
 3  if ( M(g).isPolytomy() and
         ( g = root(T_G) or M(g) ≠ M(p(g)) ) )
 4    // compute costs for a PCC
 5    Global component_root = g
 6    // compute costs
 7    ProcessComponent( g )
 8    // add losses using costs
 9    AddCombinedLosses( g, C(M(g)) \ N(g) )
10  else
11    // continue searching for PCCs
12    CombinedLosses( l(g) )
13    CombinedLosses( r(g) )

Ñ(g)
14  if ( g = component_root ) return N(g)
15  else return N̂(g)

ProcessComponent( g )
16  if ( g = component_root or M(g) = M(p(g)) )
17    ProcessComponent( l(g) )
18    ProcessComponent( r(g) )
19  else
20    component_leaf = true
21  foreach f ∈ Pow(C(M(component_root)) \ Ñ(g))
22    if ( component_leaf and f = ∅ )
23      Γ(g, f) = 0
24    else if ( component_leaf )
25      Γ(g, f) = 1
26    else
27      Γ(g, f) = CalculateCost(g, f)
```

(*continued*)

Algorithm 3. *(Continued)*

```
CalculateCost( g, f_in )
28  k* = ∞
29  k^m = ∞
30  if ( g is a Required Duplication )
31    foreach s ∈ f_in
32      f_out = f_in \ s
33      k = Γ_mem(g, f_out)
34      if ( k < k^m )
35        k^m = k
36        k* = k + 1
37        ϒ(g, f_in) = empty_list()
38        ℋ_mem(g, f_in) = 0
39        ℋ(g, f_in) = 0
40      if ( k = k^m )
41        ϒ_mem(g, f_in).addAll( ϒ_mem(g, f_out) )
42        ℋ_mem(g, f_in) = ℋ_mem(g, f_in) + ℋ_mem(g, f_out)
43        ℋ(g, f_in) = ℋ_mem(g, f_in)
44        ϒ(g, f_in) = ϒ_mem(g, f_in)
45    k = Σ_{c∈C(g)}( Γ( c, (f_in ∪ Ñ(g)) \ Ñ(c) ) )
46    f_left = (f_in ∪ Ñ(g)) \ Ñ(l(g))
47    f_right = (f_in ∪ Ñ(g)) \ Ñ(r(g))
48    // no losses here, passing on all of f_in
49    if ( k < k* )
50      k* = k
51      ϒ(g, f_in) = empty_list()
52      ℋ(g, f_in) = 0
53    if ( k = k* )
54      ϒ(g, f_in).add( f_left, f_right )
55      ℋ(g, f_in) = ℋ(g, f_in) +
              ℋ(l(g), f_left) · ℋ(r(g), f_right)
56    if ( k < k^m )
57      k^m = k
58      ϒ_mem(g, f_in) = empty_list()
59      ℋ_mem(g, f_in) = 0
60    if ( k = k^m )
61      ϒ_mem(g, f_in).add( f_left, f_right )
62      ℋ_mem(g, f_in) = ℋ_mem(g, f_in) +
              ℋ_mem(l(g), f_left) · ℋ_mem(r(g), f_right)
63    Γ_mem(g, f_in) = k^m
64  else
65    foreach s ∈ f_in
66      f_out = f_in \ s
67      k = Γ_mem(g, f_out)
68      if ( k < k^m )
69        k^m = k
70        k* = k + 1
71        ϒ(g, f_in) = empty_list()
72        ℋ_mem(g, f_in) = 0
73        ℋ(g, f_in) = 0
```

(*continued*)

Algorithm 3. *(Continued)*

```
74      if ( k = k^m )
75        Υ_mem(g, f_in).addAll( Υ_mem(g, f_out) )
76        H_mem(g, f_in) = H_mem(g, f_in) + H_mem(g, f_out)
77        H(g, f_in) = H_mem(g, f_in)
78        Υ(g, f_in) = Υ_mem(g, f_in)
79    foreach f_left ∈ Pow(f_in)
80      k = Γ( l(g), f_left )
              + Γ( r(g), f_in \ f_left )
81      if ( k < k* )
82        k* = k
83        Υ(g, f_in) = empty_list()
84        H(g, f_in) = 0
85      if ( k = k* )
86        Υ(g, f_in).add( f_left, f_in \ f_left )
87        H(g, f_in) = H(g, f_in) +
                H(l(g), f_left) · H(r(g), f_in \ f_left)
88      if ( k < k^m )
89        k^m = k
90        Υ_mem(g, f_in) = empty_list()
91        H_mem(g, f_in) = 0
92      if ( k = k^m )
93        Υ_mem(g, f_in).add( f_left, f_in \ f_left )
94        H_mem(g, f_in) = H_mem(g, f_in) +
                H_mem(l(g), f_left) · H_mem(r(g), f_in \ f_left)
95    Γ_mem(g, f_in) = k^m
96  return k*

AddCombinedLosses( g, f_in )
97  // lose genes at the edge above g
98  if ( g = component_root or M(g) = M(p(g)) )
99    f_left = Υ_l(g, f_in).pickOne()
100   f_right = Υ_r(g, f_in).pickOne()
101   Losses(g) += f_in \ (f_left ∪ f_right)
102   AddCombinedLosses( l(g), f_left )
103   AddCombinedLosses( r(g), f_right )
104 else
105   Losses(g) += f_in \ N̂(g)
```

6. RELATED WORK

Variants of LCA reconciliation have previously been proposed by numerous authors, and several software packages for analyzing gene duplication histories, when both trees are binary, have been developed (Page and Charleston, 1997; Page, 1998; Zmasek and Eddy, 2001a, 2001b; Dufayard et al., 2005; Roth et al., 2005; Durand et al., 2006a). Recently, more generalized versions of LCA reconciliation, which also calculate horizontal gene transfers, have also been presented (Gorecki, 2004; Hallett and Lagergren, 2001; Hallett et al., 2004). A related problem, inferring the optimal species tree from multiple conflicting gene trees, has been studied extensively (Bansal et al., 2007; Guigó et al., 1996; Hallett and Lagergren, 2000; Ma et al., 2000; Mirkin et al., 1995; Page, 1994; Stege, 1999; Zhang, 1997) for various optimization criteria (Eulenstein et al., 1998) and has been shown to be NP-hard (Ma et al., 2000).

In this work, we have presented algorithms to reconcile binary gene trees with non-binary species trees. These algorithms determine whether a given node in the gene tree is a required duplication, a conditional duplication, or a speciation, and reconstruct all most parsimonious loss histories. To our knowledge, these results include the first algorithms for reconciliation with non-binary species trees that infer explicit event histories, as well as the first exact algorithm for determining the minimum number of losses when the species tree is non-binary.

In work on a similar theme, Berglund-Sonnhammer et al. (2006) proposed an algorithm to infer a rooted, binary gene tree given a rooted, possibly non-binary, species tree and an unrooted, possibly non-binary, gene tree as input. This method selects a rooted, binary resolution of the input gene tree by minimizing first duplications and then losses. This parsimony criterion is encoded in open form expressions for the minimum number of duplications and losses; however, no algorithm is given for calculating these quantities. In addition, the expression for losses does not determine the species in which these losses occurred or the possible assignment of these losses to edges in the gene tree. Rather, it provides an estimated number of losses and is not guaranteed to find the optimal loss assignments. A similarity between that work and the work presented here is that both methods use set-based mappings between the gene and species trees. In this work, we proposed two such mappings, N and $\hat{N}$, of size bounded above by k_S. The "M-mapping" proposed in Berglund-Sonnhammer et al. (2006) is equivalent to our N, but there is no equivalent to $\hat{N}$ in the other work. Instead, they use a set Z that is bounded by V_S and resembles the inefficient solution proposed and rejected in Section 4.

Another related problem is the reconciliation of a *non-binary gene tree* with a *binary species tree*. Like non-binary species trees, non-binary gene trees are a common occurrence. For many data sets, the signal to noise ratio is not sufficient to fully resolve the binary branching order. As with non-binary species trees, the standard LCA algorithm will also lead to strange behavior when the gene tree is non-binary. In previous work (Durand et al., 2006b), we proposed a polynomial time algorithm to reconcile a non-binary gene tree with a binary species tree, as well as resolve the polytomies in the gene tree. These algorithms are also implemented in NOTUNG. As in Section 3, we treat polytomies in a gene tree as a set of binary hypotheses. Polytomies are then resolved by extracting from this set of hypotheses, the set of binary gene trees which result in the minimum number of duplications and losses. We derive this solution with a dynamic programming algorithm that reconstructs this minimum cost set without comparing every binary resolution of the gene tree with the species tree. Alternative approaches to solving this problem have also been presented by Berglund-Sonnhammer et al. (2006) and Chang and Eulenstein (2006), but to our knowledge have not been implemented. The combined problem of reconciling non-binary gene trees with non-binary species trees remains open.

7. EMPIRICAL RESULTS

The algorithms described here have been implemented in a new version of our software tool, NOTUNG, which is publicly available on our website. Using this software, we tested our algorithms on several data sets.

First, we compared our new reconciliation algorithms to LCA reconciliation for gene families in three species groups with known polytomies: *Anolis* (Jackman et al., 1999), *Neoaves* (Walsh et al., 1999), and Auklets (Poe and Chubb, 2004). Species trees were transcribed directly from the source articles. We constructed gene trees for two gene families in *Neoaves* (cytochrome-b and globin), three families in Auklets (cytochrome-b, cytochrome oxidase 1 and NADH-6), and one family in *Anolis* (NADH-2). Sequences were downloaded from NCBI (Wheeler et al., 2005) and multiple sequence alignments were constructed with T-Coffee (Notredame et al., 2000). Phylogeny reconstruction was performed using the PHYLIP package from Felsenstein (version 3.6.1) and bootstrapped using the included SEQBOOT program. Branches with weak bootstrap support (<60%) in the globin tree from *Neoaves* were rearranged using models presented in Durand et al. (2006b).

Table 1 shows the number of leaves (l) in each tree, the size of the maximum polytomy (k_S) in each species tree, the number of duplications obtained by LCA reconciliation (B), the number of required duplications predicted by our algorithm (R), and the optimal number of losses. For each of these gene families, the exact and heuristic loss algorithms reported the same number of losses. Only one gene family,

TABLE 1. COMPARISON OF DUPLICATIONS INFERRED BY NOTUNG AND LCA RECONCILIATION

	Tree		*Dupl.s*		
Gene family	l	k_S	B	R	*Losses*
Neoaves (Poe and Chubb, 2004)	12	10	—	—	—
Cytochrome B	9	—	4	0	0
Globin	17	—	7	4	7
Auklets (Walsh et al., 1999)	5	4	—	—	—
cox1	5	—	1	0	0
Cytochrome B	5	—	2	0	0
NADH-6	5	—	2	0	0
Anolis (Jackman et al., 1999)	50	6	—	—	—
NADH-2	50	—	13	7	17

the globins, had more than one optimal loss assignment. As predicted, LCA reconciliation substantially overestimates required duplications.

Second, we tested the accuracy of our heuristic algorithm for inferring losses by comparing it with our exact algorithm. We ran NOTUNG on all full trees in TreeFam 3.0 (Li et al., 2006) with a species tree obtained from the NCBI taxonomy (Wheeler et al., 2005). Of the 1174 gene trees in this dataset, 973 correspond to a non-binary species tree. For these 973 trees, the heuristic reported the same number of losses as the exact algorithm on all but seven trees (i.e., 99.3% of trees tested). Table 2 shows the number of losses inferred by the heuristic and the exact algorithm for those seven trees. As these results show, even when the heuristic does not obtain an optimal solution, the deviation from the minimum is slight. In the worst case, it overestimated the number of losses by four in a tree with 249 losses.

Finally, we considered the running time of the loss algorithms using the same Treefam dataset. The computational complexity of Algorithm 3 is exponential in k_S, while the heuristic in Algorithm 2 is $O(|V_G| \cdot (k_S + h_S))$. The time required to reconcile all 1174 gene trees was $0'48''$ for the heuristic and $2'25''$ for the exact algorithm on a 3.2-ghz OptiPlex GX620 computer. The running time of the memoized implementation of the exact algorithm suggests that it is acceptable for species trees with $k_S \leq 15$, the maximum k_S for any tree in the Treefam dataset. (Previous tests on running time for the exact algorithm, without memoization, showed that memoization resulted in a speed-up by a factor of $\sim$20 for this dataset.) A single tree in this dataset corresponds to $k_S = 15$; the remaining 1173 trees correspond to $k_S \leq 12$. The time required to reconcile these 1173 trees using the exact algorithm is comparable to the time for the heuristic ($0'50''$ versus $0'48''$). For data sets too large for exact analysis, our accuracy tests show that the heuristic can be relied on for near-optimal results.

TABLE 2. COMPARISON OF LOSSES INFERRED BY THE HEURISTIC AND THE EXACT ALGORITHM

	Number of losses			
TreeFam ID	*Heuristic*	*Exact*	*Offset*	*Percent incorrect*
TF106001	253	249	4	1.58
TF105558	64	63	1	1.56
TF105131	281	277	4	1.42
TF105356	71	70	1	1.41
TF105186	83	82	1	1.20
TF105188	393	391	2	0.509
TF105049	608	606	2	0.329

8. DISCUSSION

In this work, we have presented novel algorithms for the reconciliation of binary gene trees with non-binary species trees. These solutions are founded on current theories of molecular evolution, especially coalescent theory (Pamilo and Nei, 1988; Takahata and Nei, 1985; Maddison, 1997; Tajima, 1983; Hudson, 1990). Our algorithms are both space and time efficient, and have practical applications. They have been implemented in a new version of our software tool, NOTUNG, which provides a graphical user interface for the in-depth study of gene families, as well high throughput capabilities. To our knowledge, these are the first algorithms for reconciling binary gene trees with non-binary species trees to obtain the history of duplications and losses in a gene family. We also present the first exact algorithm for determining the minimum number of losses when the species tree is non-binary.

Our algorithms are of immediate use to researchers using phylogenetic analysis in a broad range of biological endeavors and are promising for further algorithmic development. For example, the loss models developed in Section 5 could also be exploited in algorithms to reconstruct a species tree from a set of gene trees using loss parsimony, as proposed by Chauve et al. (2007). Our definitions of required and conditional duplications and the set-based mappings N and $\hat{N}$ provide a potential foundation for probabilistic models of non-binary reconciliation. Such methods require a correspondence between ancestral genes and ancestral species. N and $\hat{N}$ provide this information.

Several problems remain for future work. Probabilistic models would complement the parsimony framework presented here. Bayesian approaches (Arvestad et al., 2003, 2004; DeBie et al., 2006), which assume homogeneous rates, are appropriate for data sets in which duplication and loss are neutral, stochastic processes. Parsimony is better suited to data sets in which duplication and loss are rare due to selective pressure. A probabilistic framework provides a natural setting for incorporating sequence data directly into the reconciliation process, but has the disadvantage that it is both computation and data intensive. A complete phylogenetic toolkit should include both approaches.

We considered two cases in this work: binary species trees with complete lineage sorting (i.e., one binary gene tree has probability one and all others probability zero) and non-binary species trees that give rise to all binary gene trees with equal probability. In fact, these represent extremes on a continuum. A more general approach would include polytomy models that deviate from a uniform distribution, as well as models that relax the assumption of complete lineage sorting in binary trees (Pollard et al., 2006; Lyons-Weiler and Milinkovitch, 1997; Poe and Chubb, 2004; McCracken and Sorenson, 2005; Hoelzer and Melnick, 1994). This can occur if the time between divergences is short enough so that allelic diversity has not been eliminated by genetic drift. In fact, the shorter the branch lengths and the larger the effective population size, the more closely a series of rapid binary divergences will approximate a hard polytomy (Pamilo and Nei, 1988; Takahata and Nei, 1985; Maddison, 1997; Tajima, 1983; Hudson, 1990). In addition, non-binary tree models that include horizontal gene transfer as well as gene duplication and loss are needed.

Finally, reconciliation of non-binary gene trees with non-binary species trees should also be investigated. The similarity of this problem to the problem of reconstructing a species tree from many gene trees suggests that it may be NP-complete, but the hardness of the problem remains open and algorithms (or approximation algorithms) are needed. The complexity of loss inference must also be investigated. The exact algorithm presented in Section 5.2 considers all possible loss assignments and requires exponential running time. However, whether an exponential time solution is actually required is an open question.

With the availability of sequences from many closely related genomes, it is increasingly apparent that the histories of individual genes differ and that discordance between gene and species trees is common. Software tools that are sufficiently flexible to handle this situation are needed (Pollard et al., 2006). The work presented here offers this flexibility.

ACKNOWLEDGMENTS

We thank M. Hahn, S. Hartmann, T. Hladish, R. Hoberman, H.B. Nicholas, Jr., M. Sanderson, T. Vision, R. Schwartz, and S. Sridhar for helpful discussions, and R. Cintron, H.B. Nicholas, Jr., and J. Nam for providing phylogenetic trees for the experimental analysis. This work was supported by the National

Institutes of Health (NIH grant 1 K22 HG 02451-01), Pittsburgh Supercomputing Center Biomedical Computing Initiative Computational Facilities Access Grant (MCB000010P), and a David and Lucille Packard Foundation fellowship.

DISCLOSURE STATEMENT

No conflicting financial interests exist.

REFERENCES

Arvestad, L., Berglund, A., Lagergren, J., et al. 2003. Bayesian gene/species tree reconciliation and orthology analysis using MCMC. *Bioinformatics* 19, Suppl 1, i7–i15.

Arvestad, L., Berglund, A., Lagergren, J., et al. 2004. Gene tree reconstruction and orthology analysis based on an integrated model for duplications and sequence evolution. *RECOMB 2004* 326–335.

Bansal, M., Burleigh, J., Eulenstein, O., et al. 2007. Heuristics for the gene-duplication problem: a $\theta(n)$ speed-up for the local search. *Lect. Notes Bionform.* 4453, 238–252.

Bender, M., and Farach-Colton, M. 2000. Least common ancestors revisited. *Latin '00* 88–94.

Berglund-Sonnhammer, A., Steffansson, P., Betts, M., et al. 2006. Optimal gene trees from sequences and species trees using a soft interpretation of parsimony. *J. Mol. Evol.* 63, 240–250.

Blomme, T., Vandepoele, K., De Bodt, S., et al. 2006. The gain and loss of genes during 600 million years of vertebrate evolution. *Genome Biol.* 7, R43.

Bourgon, R., Delorenzi, M., Sargeant, T., et al. 2004. The serine repeat antigen SERA gene family phylogeny in *Plasmodium*: the impact of GC content and reconciliation of gene and species trees. *Mol. Biol. Evol.* 21, 2161–2171.

Chang, W.-C., and Eulenstein, O. 2006. Reconciling gene trees with apparent polytomies. *Lect. Notes Comput. Sci.* 4112, 235–244.

Chauve, C., Doyon, J.-P., and El-Mabrouk, N. 2007. Inferring a duplication, speciation and loss history from a gene tree (extended abstract). *RECOMB-CG 2007* 45–57.

Chen, K., Durand, D., and Farach-Colton, M. 2000. Notung: a program for dating gene duplications and optimizing gene family trees. *J. Comput. Biol.* 7, 429–447.

DeBie, T., Cristianini, N., Demuth, J.P., et al. 2006. Cafe: a computational tool for the study of gene family evolution. *Bioinformatics* 22, 1269–1271.

Demuth, J.P., De Bie, T., Stajich, J.E., et al. 2006. The evolution of mammalian gene families. *PLoS ONE* 1, e85.

Dufayard, J., Duret, L., Penel, S., et al. 2005. Tree pattern matching in phylogenetic trees: automatic search for orthologs or paralogs in homologous gene sequence databases. *Bioinformatics* 21, 2596–2603.

Durand, D., Halldorsson, B., and Vernot, B. 2006a. A hybrid micro-macroevolutionary approach to gene tree reconstruction. *J. Comput. Biol.* 13, 320–335.

Durand, D., Halldorsson, B., and Vernot, B. 2006b. A hybrid micro-macroevolutionary approach to gene tree reconstruction. *J. Comput. Biol.* 13, 320–335.

Ermolaeva, M., Wu, M., Eisen, J., et al. 2003. The age of the *Arabidopsis thaliana* genome duplication. *Plant Mol. Biol.* 51, 859–866.

Eulenstein, O., Mirkin, B., and Vingron, M. 1998. Duplication-based measures of difference between gene and species trees. *J. Comput. Biol.* 5, 135–148.

Goodman, M., Czelusniak, J., Moore, G., et al. 1979. Fitting the gene lineage into its species lineage, a parsimony strategy illustrated by cladograms constructed from globin sequences. *Syst. Zool.* 28, 132–163.

Gorecki, P. 2004. Reconciliation problems for duplication, loss and horizontal gene transfer. *Proc. 8th Annu. Int. Conf. CMB* 316–325.

Gu, X., Wang, Y., and Gu, J. 2002. Age distribution of human gene families shows significant roles of both large- and small-scale duplications in vertebrate evolution. *Nat. Genet.* 31, 205–209.

Guigó, R., Muchnik, I., and Smith, T. 1996. Reconstruction of ancient molecular phylogeny. *Mol. Phylogenet. Evol.* 6, 189–213.

Hahn, M.W. 2007. Bias in phylogenetic tree reconciliation methods: implications for vertebrate genome evolution. *Genome Biol.* 8, R141.

Hahn, M.W., Han, M.V., and Han, S.-G. 2007. Gene family evolution across 12 *Drosophila* genomes. *PLoS Genet.* 3, e197.

Hallett, M., and Lagergren, J. 2000. New algorithms for the duplication-loss model. In *RECOMB 2000* 138–146.
Hallett, M., and Lagergren, J. 2001. Efficient algorithms for lateral gene transfer problems. *Proc. 5th Annu. Int. Conf. Comput. Biol.* 149–156.
Hallett, M., Lagergren, J., and Tofigh, A. 2004. Simultaneous identification of duplications and lateral transfers. *Proc. 8th Annu. Int. Conf. CMB* 347–356.
Hoelzer, G., and Melnick, D. 1994. Patterns of speciation and limits to phylogenetics resolution. *Trend Ecol. Evol.* 9, 104–107.
Hudson, R. 1990. Gene genealogies and the coalescent process, 1–44. In: *Oxford Surveys in Evolutionary Bology, Volume 7*. Eds. Futuyma, D., Antonovics, J. Oxford University Press, New York.
Jackman, T., Larson, A., De Queiroz, K., et al. 1999. Phylogenetic relationships and tempo of early diversification in *Anolis* lizards. *Syst. Biol.* 48, 254–285.
Li, H., Coghlan, A., Ruan, J., et al. 2006. TreeFam: a curated database of phylogenetic trees of animal gene families. *Nucleic Acids Res.* 34, 572–580.
Li, W. 1997. *Molecular Evolution*. Sinauer Associates Inc., Sunderland, MA.
Lyons-Weiler, J., and Milinkovitch, M. 1997. A phylogenetic approach to the problem of differential lineage sorting. *Mol. Biol. Evol.* 14, 968–975.
Ma, B., Li, M., and Zhang, L. 2000. From gene trees to species trees. *SIAM J. Comput.* 30, 729–752.
Maddison, W. 1989. Reconstructing character evolution on polytomous cladograms. *Cladistics* 5, 365–377.
Maddison, W. 1997. Gene trees in species trees. *Syst. Biol.* 46, 523–536.
McCracken, K., and Sorenson, M. 2005. Is homoplasy or lineage sorting the source of incongruent mtdna and nuclear gene trees in the stiff-tailed ducks (nomonyx-oxyura)? *Syst. Biol.* 54, 35–55.
McLysaght, A., Hokamp, K., and Wolfe, K. 2002. Extensive genomic duplication during early chordate evolution. *Nat. Genet.* 31, 200–204.
Melnick, D., Hoelzer, G., Absher, R., et al. 1993. mtDNA diversity in Rhesus monkeys reveals overestimates of divergence time and paraphyly with neighboring species. *Mol. Biol. Evol.* 10, 282–295.
Mirkin, B., Muchnik, I., and Smith, T. 1995. A biologically consistent model for comparing molecular phylogenies. *J. Comput. Biol.* 2, 493–507.
Nam, J., and Masatoshi, N. 2005. Evolutionary change in the numbers of homebox genes in bilateral animals. *Mol. Biol. Evol.* 22, 2386–2394.
Notredame, C., Higgins, D., and Heringa, J. 2000. T-Coffee: a novel method for fast and accurate multiple sequence alignment. *J. Mol. Biol.* 302, 205–217.
Page, R. 1994. Maps between trees and cladistic analysis of historical associations among genes, organisms and areas. *Syst. Biol.* 43, 58–77.
Page, R. 1998. GeneTree: comparing gene and species phylogenies using reconciled trees. *Bioinformatics* 14, 819–20.
Page, R., and Charleston, M. 1997. From gene to organismal phylogeny: reconciled trees and the gene tree/species tree problem. *Mol. Phylogenet. Evol.* 7, 231–240.
Page, R., and Holmes, E. 1998. *Molecular Evolution: A Phylogenetic Approach*. Blackwell Science, New York.
Pamilo, P., and Nei, M. 1988. Relationships between gene trees and species trees. *Mol. Biol. Evol.* 5, 568–583.
Paterson, A., Bowers, J., and Chapman, B. 2004. Ancient polyploidization predating divergence of the cereals, and its consequences for comparative genomics. *Proc. Natl. Acad. Sci. USA* 101, 9903–9908.
Perrière, G., Duret, L., and Gouy, M. 2000. HOBACGEN: database system for comparative genomics in bacteria. *Genome Res.* 10, 379–385.
Poe, S., and Chubb, A. 2004. Birds in a bush: five genes indicate explosive evolution of avian orders. *Evolution* 58, 404–415.
Pollard, D., Iyer, V., Moses, A., et al. 2006. Widespread discordance of gene trees with species tree in *Drosophila*: evidence for incomplete lineage sorting. *PLoS Genet.* 2, e173.
Ronquist, F. 2002. Parsimony analysis of coevolving species associations, 22–64. In: Page, R.D.M., ed., *Tangled Trees: Phylogeny, Cospeciation and Coevolution*. University of Chicago Press, Chicago.
Roth, C., Betts, M., Steffansson, P., et al. 2005. The adaptive evolution database (TAED): a phylogeny-based tool for comparative genomics. *Nucleic Acids Res.* 33, D495–D497.
Ruvinsky, I., and Silver, L. 1997. Newly indentified paralogous groups on mouse chromosomes 5 and 11 reveal the age of a T-box cluster duplication. *Genomics* 40, 262–266.
Salzburger, W., Meyer, A., Baric, S., et al. 2002. Phylogeny of the Lake Tanganyika cichlid species flock and its relationship to the Central and East African haplochromine cichlid fish faunas. *Syst. Biol.* 51, 113–135.
Searls, D. 2003. Pharmacophylogenomics: genes, evolution and drug targets. *Nat. Rev. Drug Discov.* 2, 613–623.
Sennblad, B., Schreil, E., Berglund Sonnhammer, A., et al. 2007. Primetv: a viewer for reconciled trees. *BMC Bioinform.* 8, 148.
Slowinski, J., and Page, R. 1999. How should species phylogenies be inferred from sequence data? *Syst. Biol.* 48, 814–825.

Stege, U. 1999. Gene trees and species trees: the gene-duplication problem is fixed-parameter tractable. *Lect. Notes Comput. Sci.* 1663, 288–293.

Tajima, F. 1983. Evolutionary relationship of dna sequences in finite populations. *Genetics* 105, 437–460.

Takahata, N., and Nei, M. 1985. Gene genealogy and variance of interpopulational nucleotide differences. *Genetics* 110, 325–344.

Vandepoele, K., Simillion, C., and Van de Peer, Y. 2003. Evidence that rice and other cereals are ancient aneuploids. *Plant Cell* 15, 2192–2202.

Vandepoele, K., Vos, W., Taylor, J., et al. 2004. Major events in the genome evolution of vertebrates: paranome age and size differ considerably between ray-finned fishes and land vertebrates. *Proc. Natl. Acad.Sci. USA* 101, 1638–1643.

Walsh, H., Kidd, M., Moum, T., et al. 1999. Polytomies and the power of phylogenetic inference. *Evolution* 53, 932–937.

Wang, X., Shi, X., Hao, B., et al. 2005. Duplication and DNA segmental loss in the rice genome: implications for diploidization. *New Phytol.* 165, 937–946.

Wheeler, D., Hope, R., Cooper, S., et al. 2001. An orphaned mammalian beta-globin gene of ancient evolutionary origin. *Proc. Natl. Acad. Sci. USA* 98, 1101–1106.

Wheeler, D., Barrett, T., Benson, D.A., et al. 2005. Database resources of the National Center for Biotechnology Information. *Nucleic Acids Res.* 33, D39–D45.

Zhang, L. 1997. On a Mirkin–Muchnik-Smith conjecture for comparing molecular phylogenies. *J. Comput. Biol.* 4, 177–188.

Zmasek, C., and Eddy, S. 2001a. ATV: display and manipulation of annotated phylogenetic trees. *Bioinformatics* 17, 383–384.

Zmasek, C., and Eddy, S. 2001b. A simple algorithm to infer gene duplication and speciation events on a gene tree. *Bioinformatics* 17, 821–828.

Address reprint requests to:
Dr. Dannie Durand
Department of Biological Sciences
Carnegie Mellon University
Pittsburgh, PA 15213

E-mail: durand@cmu.edu