A general framework for gene tree correction based on duplication-loss reconciliation

Nadia El-Mabrouk[1] **and Aïda Ouangraoua**[2]

1 **Département d'informatique et de recherche opérationnelle, Université de Montréal, Québec, Canada**
`mabrouk@iro.umontreal.ca`

2 **Département d'informatique, Université de Sherbrooke, Québec, Canada**
`aida.ouangraoua@usherbrooke.ca`

Abstract

Due to the key role played by gene trees and species phylogenies in biological studies, it is essential to have as much confidence as possible on the available trees. As phylogenetic tools are error prone, it is a common task to use a correction method for improving an initial tree. Various correction methods exist. In this paper we focus on those based on the Duplication-Loss reconciliation model. The polytomy resolution approach consists in contracting weakly supported branches and then refining the obtained non-binary tree in a way minimizing a reconciliation distance with the given species tree. On the other hand, the supertree approach takes as input a set of separated subtrees, either obtained for separared orthology groups or by removing the upper branches of an initial tree to a certain level, and amalgamating them in an optimal way preserving the topology of the initial trees. The two classes of problems have always been considered as two separate fields, based on apparently different models. In this paper we give a unifying view showing that these two classes of problems are in fact special cases of a more general problem that we call LabelGTC, whose input includes a 0-1 edge-labelled gene tree to be corrected. Considering a tree as a set of triplets, we also formulate the TripletGTC Problem whose input includes a set of gene triplets that should be preserved in the corrected tree. These two general models allow to unify, understand and compare the principles of the duplication-loss reconciliation-based tree correction approaches. We show that LabelGTC is a special case of TripletGTC. We then develop appropriate algorithms allowing to handle these two general correction problems.

1998 ACM Subject Classification Biology and genetics

Keywords and phrases gene tree correction – supertree – polytomy — reconciliation – phylogeny

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.23

1 Introduction

Studying the functional specificities of gene copies, such as their role in metabolic pathways of interest, usually requires a trusted gene tree. However, for various reasons related to the specificities of phylogenetic software, the considered evolutionary models or errors in the multiple alignments, constructed trees are usually not fully satisfactory. Consequently, most tree construction methods integrate measures of statistical support obtained by bootstrapping or jackknifing [3], reflecting the confidence we have on the prediction. A strong support on a branch reflects a strong support on the clade (in case of a rooted tree) or the bipartition (in case of an unrooted tree) represented by this branch. Results coming out from bioinformatics pipelines should then be analyzed in light of this uncertainty in the considered trees.

Alternatively, trees may be corrected before carrying on with the biological analysis. Due to the fact that standard gene tree databases, such as Ensembl [15], are known to be error prone, gene tree correction methods are frequently used upstream to obtain better trees for gene families (c.f. for example NOTUNG [2], TreeFix [16], ProfileNJ[10], MowgliNNI [9], ecceTERA [4], MRL [8]). Most are based on minimizing a reconciliation distance with a given species tree, some including horizontal gene transfer. In this paper, we restrict ourselves to the duplication-loss reconciliation distance that we simply call the reconciliation distance. Moreover, we focus on polytomy-based and supertree-based correction methods that cover a large set of correction methods, although not all (for example, TreeFix [16] relies on exploring a space surrounding an initial tree, and cannot be categorized as a polytomy or supertree-based correction method.)

Polytomy-based correction methods rely on contracting branches with weak support, leading to a non-binary tree, and then resolving multifurcated nodes (polytomies) in a binary way minimizing the reconciliation distance with the species tree [2, 5, 10, 12, 17]. The most efficient algorithm for resolving a non-binary tree is linear in the size of the tree [5, 17]. Such a polytomy resolution approach preserves the subtrees in terms of topology and gene content. In other words, the exhibited monophily of input gene clusters is not challenged by a polytomy resolution method.

Other methods rather stand on amalgamating “trusted” partial trees into a single one for the whole family [8, 14]. Such partial trees may be obtained by constructing them independently for partial sets of orthologs, or by removing weakly supported branches of an initial tree. In [6, 7], we have formalized this approach in terms of a supertree method for gene trees. The defined SuperGeneTree (SGT) problem consists in constructing, from a set of partial trees, a tree containing them all and minimizing the reconciliation distance. A simplest version considering the duplication distance has been shown NP-hard [6]. Conceptually, the supertree-based correction method is more general than the polytomy-based one, as only the topology of initial trees should be preserved in the former case, while the latter requires also to preserve the clades. Although it may be relevant to challenge the monophiletic nature of input partial gene trees, SGT may lead to a drastic reorganization giving rise to a tree grouping genes that are far apart in the original tree. To avoid this problem, we also introduced the Triplet Respecting Supergenetree (TRS) problem [7], asking for a supertree displaying all input subtrees, while preserving the topology of any triplet of genes taken from three different subtrees (clades).

The polytomy-based and supertree-based models of gene tree correction have been developed separately, considering separate assumptions and constraints. Some assumptions are actually questionable such as the one considering upper branches as the ones that should be removed in the case of a polytomy resolution, or alternatively kept in the case of TRS. In fact, support can be strong or weak on any branch of the tree. Moreover, in the absence of a unifying model, the conservative or permissive nature of each method with respect to another one can only be tested empirically. Here, we show that all these methods can in fact be considered in a unifying way, as special cases of a more general gene tree correction problem taking as input a 0-1 edge-labelling derived from edge statistical supports.

In Section 3, we introduce the Label Respecting GeneTreeCorrection (LabelGTC) Problem whose input includes a 0-1 edge-labelled gene tree to be corrected. We show that the polytomy related and supertree related correction problems are all special cases of this problem. In Section 4, we then define Triplet Respecting GeneTreeCorrection (TripletGTC) Problem, a more general problem whose input includes a set of gene triplets that should be preserved in the corrected tree. We show that LabelGTC is a special case of

TripletGTC. As these problems include the SGT problem as a sub-problem, they are both NP-hard for the duplication distance. In Section 5, we then exhibit a recursive algorithm for LabelGTC and show how it can be used to define a heuristic algorithm for TripletGTC. Finally, a variant of the LabelGTC Problem, allowing for an extended labelling, is presented in Section 6. Developed algorithms have the same exponential time-complexity as the one previously developed for the SGT problem [7], that is a particular case of the new problems studied here. All missing proofs are given in Appendix.

2 Preliminaries on gene tree correction methods

Notations on trees: All considered trees are rooted. A tree is *binary* if each internal node (all nodes except the leaves) has exactly two children, and *non-binary* otherwise. When not mentioned explicitly, all trees are considered binary (except for the polytomy resolution problem; we mention it explicitly in this case).

A tree T with leafset $\mathcal{L}(T) = X$ is called a tree for X. If X is a set of species, then T is a *species tree*. We denote by $V(T)$ the node set and by $E(T)$ the edge set of T. An edge of $E(T)$ is written as a pair (x, y) of two adjacent nodes, where x is the closest to the root. A node x is *an ancestor* of y if it is on the path from y to the root (excluding y). In this case, y is called *a descendant* of x. Similarly, an edge (x', y') is *an ancestor* of an edge (x, y) if it is on the path from (x, y) to the root. Given a node x, $T[x]$ is the subtree of T rooted at x and $\mathcal{L}(x)$ the leafset of $T[x]$. Two subtrees $T[x]$ and $T[y]$ are *separated* in T iff $x \neq y$ and none of x or y is an ancestor of the other. It follows that $\mathcal{L}(x) \cap \mathcal{L}(y) = \emptyset$.

The *lowest common ancestor* of $L' \subseteq \mathcal{L}(T)$, denoted $lca_T(L')$, is the ancestor common to all leaves in L' that is the most distant from the root. $T|_{L'}$ is the tree with leafset L' obtained from the subtree of T rooted at $lca_T(L')$ by removing all leaves that are not in L', and then all internal nodes of degree 2, except the root of this tree. Let T' be a tree such that $\mathcal{L}(T') = L' \subseteq \mathcal{L}(T)$. We say that T *displays* T' iff $T|_{L'}$ is label-isomorphic to T'. In this case, we also say that T *preserves* the topology of T'.

A set $\mathcal{T}$ of trees, with possibly overlapping leafsets, is *consistent* if there is a tree T' on the union of their leafsets displaying them all. Such a tree T' is called a *supertree* for $\mathcal{T}$. For example, in Figure 1, the tree T' is a supertree for the set of subtrees of T circled in blue color.

Gene tree and reconciliation: A *gene family* is a set of genes Γ accompanied with a *mapping function* $s : \Gamma \to \Sigma$ from each gene to its corresponding species in a set of species Σ. Consider a gene family Γ where each gene $x \in \Gamma$ belongs to a species $s(x)$ of Σ. The evolutionary history of Γ can be represented as a *gene tree* T for Γ. Each internal node of T refers to an ancestral gene at the moment of an event, either speciation (*Spec*) or duplication (*Dup*). Let S be a species tree for Σ. The mapping s is extended to be defined from $V(T)$ to $V(S)$ as follows: if x is an internal node of T, then $s(x) = lca_S(\{s(x') : x' \in \mathcal{L}(x)\})$.

When the type of event is known for each internal node, the gene tree T is said *labelled*. Formally, a *labelled gene tree* for Γ is a pair (T, ev_T), where T is a tree for $\mathcal{L}(T) = \Gamma$, and $ev_T : V(T) \setminus \mathcal{L}(T) \to \{Dup, Spec\}$ is a function labelling each internal node of T as a duplication or a speciation node.

The *lca-reconciliation* (or simply *reconciliation* for short) of a gene tree T with a species tree S is the labelled tree (T, ev_T) obtained by labelling each internal node x of T with children x_l and x_r as *Spec* iff $s(x_l)$ and $s(x_r)$ are separated in S, and as *Dup* otherwise. The mapping function s and the node labelling also induce gene loss events on branches of the gene tree (see Figures 1 and 2 for examples of lca-reconciliations of gene trees with species

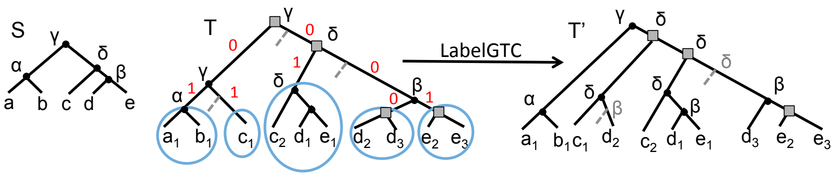

Figure 1 Left. A species tree S for $\Sigma = \{a, b, c, d, e\}$, a reconciled 0-1 edge-labelled gene tree T for $\Gamma = \{a_1, b_1, c_1, c_2, d_1, d_2, d_3, e_1, e_2, e_3\}$ where each leaf x_i denotes a gene belonging to species x, and a covering set $\mathcal{T}$ of subtrees for T indicated by circles around each subtree. Square nodes are duplications and circular nodes are speciations. Internal nodes are labelled according to corresponding ancestral species in S. Dotted lines are losses. **Right.** A supertree for $\mathcal{T}$ of minimum reconciliation cost (cost of 5) respecting the edge labelling of T .

trees, and the induced loss events). The *reconciliation cost* of a labelled tree (T, ev_T) is its number of duplication nodes and induced loss events.

We end this section with a final definition. Let T and T' be two trees on Γ. If (x, y) is an edge of $E(T)$ and there is an edge (x', y') in $E(T')$ such that $\mathcal{L}(y) = \mathcal{L}(y')$, we say that T' *preserves* the edge (x, y) of T.

3 A unifying view on gene tree correction problems

In the remaining of the paper, S is a species tree on a species set Σ and T is a gene tree for a gene set Γ.

The correction problem asks for a "better tree" T' for Γ according to the reconciliation cost. The various versions of the problem differ on the flexibility we have on modifying T. Which parts of T should be preserved ? The most natural way to do is to preserve all well supported branches, according to a given statistical support, and be allowed to modify all weakly supported branches. Notice that the support on a branch (x, y) reflects the confidence we have on the fact that $\mathcal{L}(y)$ represents a separate clade in the gene family.

The underlying representation is a 0-1 edge labelling of T edges, where 0 indicates a low support and 1 a high support according to a certain threshold.

In addition, if the tree T contains a set of separated subtrees whose topologies are "trusted", they should be considered as an additional parameter. Such trusted topologies may, for instance, be those obtained separately for different orthology groups agreeing with the species tree, and used to build T.

Accordingly, the most general gene tree correction problem is formulated below, where a *covering set of subtrees* $\mathcal{T}$ for T is a set of separated subtrees of T, $\mathcal{T} = \{T[x_1], T[x_2], \ldots, T[x_n]\}$ such that $\cup_{i=1}^{n} \mathcal{L}(x_i) = \mathcal{L}(T)$, and a 0-1 edge labelling for T is a function l defined from the set of edges $E(T)$ to $\{0,1\}$. In the following formulation, edge labels are ignored for the trees of $\mathcal{T}$ (see Figure 1 for an illustration of the LabelGTC Problem).

Label Respecting GeneTreeCorrection (LabelGTC) Problem:
Input: A species tree S, a gene tree T, a covering set of trees $\mathcal{T}$ for T and a 0-1 edge labelling l for T.
Output: A supertree T' for $\mathcal{T}$ of minimum reconciliation cost such that: if $(x, y) \in E(T) \setminus E(\mathcal{T})$ is such that $l(x, y) = 1$, then there is an edge (x', y') in $E(T')$ such that $\mathcal{L}(y) = \mathcal{L}(y')$.

Notice that if no information on "trusted" separated subtrees is available, then each tree of $\mathcal{T}$ is simply restricted to a leaf of T, in which case $\mathcal{T}$ simply refers to the leafset of T.

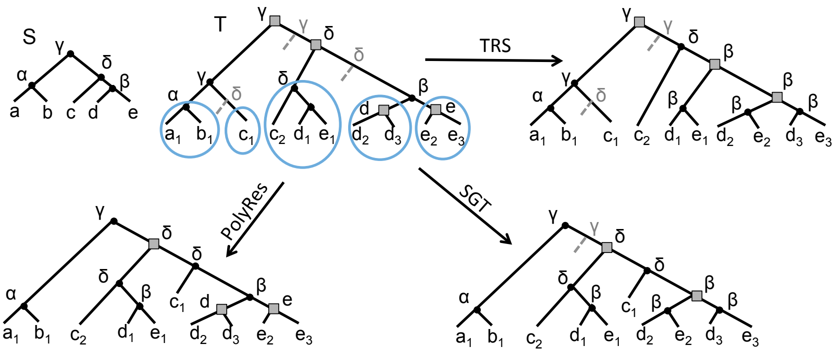

Figure 2 **Top left.** Species tree S for $\Sigma = \{a, b, c, d, e\}$, gene tree T for $\Gamma = \{a_1, b_1, c_1, c_2, d_1, d_2, d_3, e_1, e_2, e_3\}$ with a covering set $\mathcal{T}$ of subtrees for T as in Figure 1 (without the 0-1 labelling of edges). **Bottom left.** A polytomy resolution supertree for $\mathcal{T}$ of minimum reconciliation cost (cost of 3). **Bottom right.** A supertree for $\mathcal{T}$ of minimum reconciliation cost (cost of 3). **Top right.** A triplet respecting supertree for $\mathcal{T}$ of minimum reconciliation cost (cost of 5). Note that the three optimal supertrees for the TRS, SGT and PolyRes problems differ from the optimal supertree for the LabelGTC problem depicted in Figure 1, because the implicit 0-1 edge-labellings differ from the one in Figure 1.

The above formulation is not the one actually considered in the literature. Some special cases of the LabelGTC problem where the input covering set of trees $\mathcal{T}$ is the leafset of T were considered in [2, 5, 10, 12] under the name of Polytomy Resolution problems, and in [6, 7, 8, 14] under the name of Supertree problems. In the following paragraphs, we recall these polytomy related and supertree related correction problems (see Figure 2 for an illustration of the problems). We will show later that they are all special cases of the general LabelGTC formulation.

The polytomy resolution problem:

The general version of the problem consists in contracting all weakly supported internal branches of the input gene tree T, leading to a non-binary tree denoted by T^{nb}, and then finding a binary refinement of T^{nb} minimizing the reconciliation cost. Formally, given a non-binary gene tree T^{nb}, a binary tree T' is a *binary refinement* of T^{nb} if for any node x of T^{nb}, there exists a node x' in T' such that $\mathcal{L}(x) = \mathcal{L}(x')$.

The simplest form of a non-binary tree is a *polytomy* defined as a set of leaves $\mathcal{L}$, all being adjacent to the root. Given a non-binary gene tree T^{nb}, it has been shown that a refinement of minimum cost for T^{nb} can be obtained by a depth-first procedure iteratively solving each polytomy $T[x]$, for each internal non-binary node x of T^{nb}. This is the reason for the name given to the general problem formulated below. The restriction to a single polytomy is formulated afterwards. It is also required in the main Theorems 1 and 6 of the paper.

Mutiple Polytomy Resolution (M-PolyRes) Problem:
Input: A species tree S and a 0-1 edge-labelled gene tree T.
Output: A binary refinement of T^{nb} minimizing the reconciliation distance.

As stated above, the simplest form of the problem is a single polytomy. It consists in having a single non-binary node in T^{nb}, the root, such that the subtrees rooted at the children of the root are "trusted" partial trees that should remain rooted subtrees of the final tree (see the tree obtained from PolyRes in Figure 2).

Polytomy Resolution (PolyRes) Problem:
Input: A species tree S, a gene tree T and a covering set of trees $\mathcal{T}$ for T.
Output: A supertree T' for $\mathcal{T}$ of minimum reconciliation cost such that for any tree $T_i \in \mathcal{T}$, $T'_{|\mathcal{L}(T_i)} = T_i$.

The supertree resolution problem:

The above formulation of the polytomy resolution problem is a special case of the more general supertree problem, where the constraint of preserving the monophily of input “trusted” partial trees is relaxed. In [6], the SuperGenetree correction problem is formulated as follows.
SuperGenetree (SGT) Problem:
Input: A species tree S, a gene tree T and a covering set of trees $\mathcal{T}$ for T.
Output: A supertree T' for $\mathcal{T}$ of minimum reconciliation cost.

The triplet-respecting supertree problem:

To avoid having a supertree grouping genes that are far apart in the original tree, we introduced, in [7], an alternative problem allowing to restrict the output space to supertrees preserving the topology of any triplet of genes taken from three different input subtrees of $\mathcal{T}$. The triplet-based constrained supertree problem is the following.

Triplet-Respecting SuperGeneTree (TRS) Problem:
Input: A species tree S, a gene tree T and a covering set of trees $\mathcal{T}$ for T.
Output: A supertree T' for $\mathcal{T}$ of minimum reconciliation cost respecting the following property: for any triplet (a, b, c) where a, b and c are genes of Γ being leaves of three different trees of $\mathcal{T}$, $T'_{|\{a,b,c\}} = T_{|\{a,b,c\}}$.

For example, the tree which is a solution of the SGT Problem in Figure 2 is not a solution of the TRS problem as the triplet (a_1, c_1, c_2), where each gene belongs to a separate subtree of the covering set $\mathcal{T}$ of T, has the topology $(a_1, (c_1, c_2))$ in this tree while it has the topology $((a_1, c_1), c_2)$ in T.

A unifying view:

The following Theorem shows that the polytomy related and supertree related problems are in fact special cases of the general LabelGTC problem. Given a covering set of subtrees $\mathcal{T}$ for T, we call a *terminal edge* an edge of $E(T) \setminus E(\mathcal{T})$ which is adjacent to a tree of $\mathcal{T}$. All other edges of $E(T) \setminus E(\mathcal{T})$ are called *non-terminal edges* (see Figure 3 for an illustration).

▶ **Theorem 1.** *Let T be a 0-1 edge-labelled gene tree and $\mathcal{T}$ be a covering set for T. Then the* LabelGTC *Problem is reduced to:*

1. *the* M-PolyRes *Problem if $\mathcal{T} = \mathcal{L}(T)$; Otherwise:*
2. *the* PolyRes *Problem if all non-terminal edges are labelled* 0*, and all terminal edges are labelled* 1*;*
3. *the* SGT *Problem if all non-terminal and terminal edges are labelled* 0*;*
4. *the* TRS *Problem if all non-terminal edges are labelled* 1*, and all terminal edges are labelled* 0.

Proof. The problem LabelGTC on $\{T, \mathcal{L}(T)\}$ is reduced to:

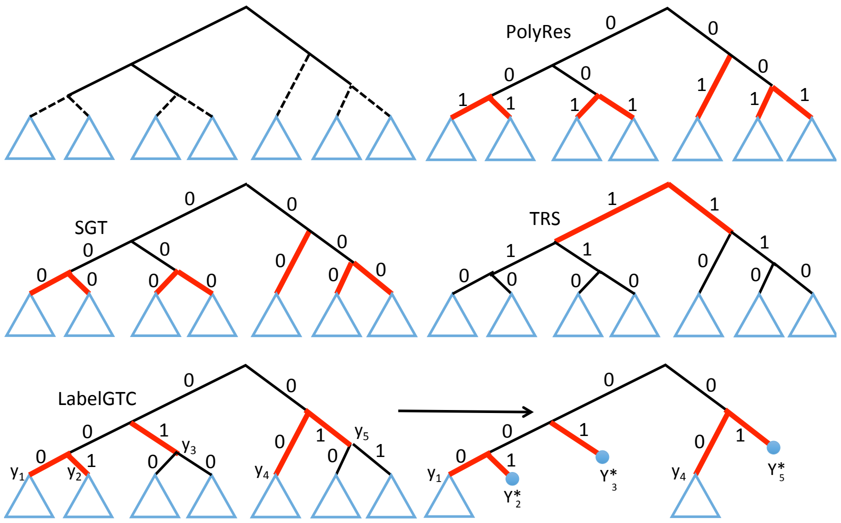

Figure 3 Top left. A gene tree T with a covering set $\mathcal{T}$ composed of 7 subtrees indicated as triangles. The set $E(T) \setminus E(\mathcal{T})$ contains 7 terminal edges (dotted lines) and 5 non-terminal edges (solid lines). Four 0-1 edge labelling corresponding to: **Top right.** the PolyRes problem ; **Middle left.** the SGT problem ; **Middle right.** the TRS problem ; **Bottom left.** a general input of the LabelGTC problem. In all four cases, the largest covering set of edges of $E(T)$ that have no ancestral edge labelled 1 are indicated with thicker red lines. **Bottom right.** The tree T^* built at step 2.b.) of the LabelGTC algorithm (Theorem 8) for the general input (Bottom left).

1. M-PolyRes if $\mathcal{T} = \mathcal{L}(T)$, because there is a bijection between the edges of $E(T)$ labelled 1 and the set of nodes (excluding the root node) of the non-binary tree T^{nb} obtained from T by contracting all edges labelled 0 : an edge (x, y) of T labelled 1 corresponds to a node y' of T^{nb} such that $\mathcal{L}(y) = \mathcal{L}(y')$. So a tree T' preserves the edges of $E(T)$ labelled 1 iff it is a binary refinement of T^{nb}.
2. PolyRes if all non-terminal edges are labelled 0 and all terminal edges are labelled 1, because a supertree T' of $\mathcal{T}$ preserves the terminal edges of $E(T)$ iff for any tree $T_i \in \mathcal{T}$, $T'_{|\mathcal{L}(T_i)} = T_i$.
3. SGT if all non-terminal and terminal edges are labelled 0, because the set of edges to be conserved by LabelGTC is empty in this case.
4. TRS if all non-terminal edges are labelled 1 and all terminal edges are labelled 0, because any triplet (a, b, c) of genes belonging to three different trees of $\mathcal{T}$ can be associated to a non-terminal edge of $E(T)$ as follows: suppose that a and $y = lca(b, c)$ are separated w.l.o.g. and let x be the parent node of y, then the edge (x, y) of T is a non-terminal edge of $E(T)$ because b and c belong to two different trees of $\mathcal{T}$ and (a, b, c) belongs to $Trp(x, y)$ (Definition 2). Now, it suffices to notice that a supertree T' for $\mathcal{T}$ preserves a non-terminal edge (x, y) of T iff it preserves the topology of all triplets of $Trp(x, y)$.

◀

4 Relating gene tree correction problems to triplets

So far, the TRS Problem is the only one that was defined in terms of triplets. However, as a rooted tree is fully determined by the topology of its set of leaf triplets, TRS can be seen as a special case of a very natural general gene tree correction problem, that we formulate below.

We first need to introduce some definitions. Generalizing the notion used for the TRS problem, a *triplet of genes* is a triplet (a, b, c) of distinct genes of Γ. By convention and without loss of generality, we consider that a and $lca(b, c)$ are separated in T. Let T and T' be two trees for Γ and Trp be a set of triplets. We say that T' is *triplet respecting* for Trp as compared to T iff T' displays the same topology as T for each triplet of Trp. The general gene tree correction problem is formulated as follows.

TRIPLET-RESPECTING GENETREECORRECTION (TRIPLETGTC) PROBLEM:
Input: A species tree S, a gene tree T, a covering set of trees $\mathcal{T}$ for T and a set Trp of triplets;
Output: A supertree T' for $\mathcal{T}$ of minimum reconciliation cost respecting Trp.

The set Trp can be restricted to triplets with genes belonging to at least two different trees of $\mathcal{T}$, as the other triplets are necessarily displayed by a supertree for $\mathcal{T}$. We call TRIPLETGTC$_{leaves}$ the TRIPLETGTC problem in the special case where $\mathcal{T} = \mathcal{L}(T)$.

The following theorem makes the link between the LABELGTC problem and the TRIPLETGTC problem. First, we formally define the set of triplets associated to an edge and that associated to a rooted subtree.

▶ **Definition 2.** Let (x, y) be an edge of $E(T)$. The set of triplets $Trp(x, y)$ contains all the triplets (a, b, c) such that a and y are separated and $lca(b, c) = y$.

▶ **Definition 3.** Let T_i be a subtree of T. The set of triplets $Trp(T_i)$ contains all the triplets (a, b, c) such that a, b and c are leaves of T_i.

For example, in the gene tree T depicted in Figure 1, if (x, y) is the non-terminal edge such that $\mathcal{L}(y) = \{d_2, d_3, e_2, e_3\}$, then $Trp(x, y) = \{(a, b, c) \,|\, a \in \Gamma \setminus \{d_2, d_3, e_2, e_3\} \, and \, (b, c) \in \{d_2, d_3\} \times \{e_2, e_3\}\}$, and $= Trp(T[y]) = \{(d_2, d_3, e_2), (d_2, d_3, e_3), (d_2, e_2, e_3), (d_3, e_2, e_3)\}$.

Note that a tree T' for $\mathcal{L}(T)$ preserves an edge (x, y) of $E(T) \setminus E(\mathcal{T})$ iff it preserves the topology of all triplets in $Trp(x, y)$. Similarly, it preserves the topology of a subtree T_i of T iff it preserves the topology of all triplets in $Trp(T_i)$.

▶ **Theorem 4.** *Let T be a 0-1 edge-labelled gene tree, $\mathcal{T}$ be a covering set for T and Trp be a set of triplets. Then, the* TRIPLETGTC *Problem is reduced to the* LABELGTC *Problem iff* $Trp = \{Trp(x, y) \mid (x, y) \in E(T) \setminus E(\mathcal{T}) \, and \, l(x, y) = 1\}$.

We now make the link between the various gene tree correction problems and the TRIPLETGTC$_{leaves}$ problem by analyzing the output of TRIPLETGTC$_{leaves}$ depending on the input set of triplets Trp. With respect to the set of initial subtrees $\mathcal{T}$, a triplet (a, b, c) of genes can either be included in a single subtree or distributed among two or three subtrees. We formally define these three possibilities of triplet-respecting trees in the next definition.

▶ **Definition 5.** Let T be a gene tree and $\mathcal{T}$ be a covering set for T. Each of the following sets of triplets contains all the triplets (a, b, c) where a, b and c are disjoint genes satisfying the corresponding property:

- $Trp1$: a, b, c all belong to the same tree of $\mathcal{T}$;
- $Trp2$: a, b, c belong to two different trees of $\mathcal{T}$;
- $Trp3$: a, b, c all belong to different trees of $\mathcal{T}$.

The following theorem extends the result of Theorem 4.

▶ **Theorem 6.** *Let T be a tree, $\mathcal{T}$ be a covering set of subtrees for T and Trp be a set of triplets. Then the* TRIPLETGTC$_{leaves}$ *Problem is reduced to:*

1. *the Identity if $Trp = Trp1 \cup Trp2 \cup Trp3$ (no modification of the input tree); Otherwise:*
2. *if $Trp = Trp1 \cup Trp2$, the* PolyRes *Problem;*
3. *if $Trp = Trp1 \cup Trp3$, the* TRS *Problem;*
4. *if $Trp = Trp2 \cup Trp3$, the* M-PolyRes *Problem with:*
   - **a.** *all non-terminal and terminal edges labelled* 1 *and,*
   - **b.** *all other edges (in $E(\mathcal{T})$) labelled* 0*;*
5. *if $Trp = Trp1$, the* SGT *Problem;*
6. *if $Trp = Trp2$, the* M-PolyRes *Problem with:*
   - **a.** *all terminal edges labelled* 1*;*
   - **b.** *all other edges (non-terminal and in $E(\mathcal{T})$) labelled* 0*.*
7. *if $Trp = Trp3$, the* M-PolyRes *Problem with:*
   - **a.** *all non-terminal edges labelled* 1 *and,*
   - **b.** *all other edges (terminal and in $E(\mathcal{T})$) labelled* 0*.*

5 Algorithm for the LabelGTC Problem

In this section, we describe an algorithm for the LabelGTC problem in the general case of a 0-1 edge labelling that does not correspond to a pre-defined gene tree correction method, as pointed out by Theorem 1. We will show later that it leads to a heuristic algorithm for the more general TripletGTC problem.

The idea behind the algorithm for reconstructing the new tree T' from the input tree T is the following. For any edge (x, y) in $E(T) \setminus E(\mathcal{T})$ such that $l(x, y) = 1$, by definition of the LabelGTC Problem, there exists a node y' of T' such that $\mathcal{L}(y') = \mathcal{L}(y)$. So the subtree $T'[y']$ of T' for the subset $\mathcal{L}(y)$ can first be constructed independently from the remaining of the tree, and then grafted at the appropriate location in a way minimizing the reconciliation cost. This leads to a recursive algorithm reconstructing and amalgamating iteratively, in a bottom-up order, the subtrees of T' for subsets $\mathcal{L}(y)$ corresponding to the edges (x, y) in $E(T) \setminus E(\mathcal{T})$ verifying $l(x, y) = 1$. The root $r(T)$ of T is associated to a dummy edge (s, r) such that $l(s, r) = 1$.

A *covering set of edges* for T is a set of separated edges $\mathcal{E} = \{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\}$ such that $\mathcal{E} \subseteq E(T) \setminus E(\mathcal{T})$ and $\cup_{i=1}^{n} \mathcal{L}(y_i) = \mathcal{L}(T)$.

Lemma 7 describes a property of covering sets that will be useful for a formal description of the algorithm solving LabelGTC.

▶ **Lemma 7.** *Let (x, y) be an edge in $E(T) \setminus E(\mathcal{T})$ such that $l(x, y) = 1$, and $\mathcal{E}$ be the largest covering set of edges of $E(T[y])$ that have no ancestral edge in $E(T[y])$ labelled* 1*. Then, any edge of $\mathcal{E}$ labelled* 0 *is a terminal edge.*

Given an edge (x, y) in $E(T) \setminus E(\mathcal{T})$ such that $l(x, y) = 1$, to compute the subtree $T'[y']$ of T' for $\mathcal{L}(y)$, first we look for the largest covering set of edges $\mathcal{E}$ for $T[y]$ such that any edge in $\mathcal{E}$ has no ancestral edge in $E(T[y])$ labelled 1 (see Figure 3 for illustration). Next, we distinguish two possible cases. If all edges in $\mathcal{E}$ are labelled 0, then $T'[y']$ can be obtained by applying the SGT algorithm [7] (designed as SGT in Algorithm 1) on the set of subtrees of $\mathcal{T}$ belonging to $T[y]$. Otherwise, $\mathcal{E}$ contains edges that are labelled 1. In this case, we compute each of the subtrees of $T'[y']$ corresponding to these edges labelled 1, yielding a set of subtrees $\mathbb{T}'$ and then we build $T'[y']$ using the SGT algorithm again with the constraint that the trees in $\mathbb{T}'$ should remain unmodified.

Algorithm 1 *LabelGTC*$(S, T, \mathcal{T}, l)$

```
$\mathcal{E} = \{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\}$ is the largest covering set of edges for $T$ in $E(T) \setminus E(\mathcal{T})$
that have no ancestral edge labelled 1.
(Stop condition)
if all edges of $\mathcal{E}$ are labelled 0 then
  return ($SGT(S, T, \mathcal{T})$)
end if
(Iterative step)
for $(x_i, y_i) \in \mathcal{E}$ such that $l(x_i, y_i) = 1$ do
  $T'[y'_i] = LabelGTC(S, T[y_i], \mathcal{T}_{|\mathcal{L}(y_i)}, l_{|E(T[y_i])})$;
end for
$T^*$ is obtained from $T$ by contracting each subtree $T[y_i]$ such that $(x_i, y_i) \in \mathcal{E}$ and
$l(x_i, y_i) = 1$ to a single leaf node $y_i^*$;
$\mathcal{T}^*$ is the set of separated subtrees $\{T[y_i] : (x_i, y_i) \in \mathcal{E}\ and\ l(x_i, y_i) = 0\} \cup \{y_i^* : (x_i, y_i) \in$
$\mathcal{E}\ and\ l(x_i, y_i) = 1\}$ of $T^*$;
$T'^* = SGT(S, T^*, \mathcal{T}^*)$;
return (the tree obtained from $T'^*$ by replacing each leaf node $y_i^*$ by $T'[y'_i]$);
```

Algorithm 2 *LabelGTC*$(S, T, \mathcal{T}, l)$

```
$\mathcal{E} = \{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\}$ is the largest covering set of edges for $T$ in $E(T) \setminus E(\mathcal{T})$
that have no ancestral edge labelled 1;
SPECIAL CASES:
if $\mathcal{T} = \mathcal{L}(T)$ then
  $T'$ is a multi-polytomy tree obtained from $T$
  $\mathcal{T} = MultiPolyRes(S, T')$
else if all terminal edges are labelled 1 and all non-terminal edges are labelled 0 then
  $T'$ is a polytomy obtained from $T$
  $\mathcal{T} = PolyRes(S, T')$
else if all terminal and non-terminal edges are labelled 0 then
  $\mathcal{T} = SGT(S, \mathcal{T})$
else if all terminal edges are labelled 0 and all non-terminal are labelled 1 then
  $\mathcal{T} = MinTRS(S, T, \mathcal{T})$
else
  ITERATIVE STEP:
  for $(x_i, y_i) \in \mathcal{E}$ such that $l(x_i, y_i) = 1$ do
    $T[y_i] = LabelGTC(S, T[y_i], \mathcal{T}_{|\mathcal{L}(y_i)}, l_{|E(T[y_i])})$;
    if no special case detected then
      $\mathcal{T}^*$ is the set of separated subtrees $\{T[y_i] : (x_i, y_i) \in \mathcal{E}\}$;
      The clades in $\{T[y_i] : (x_i, y_i) \in \mathcal{E}\ and\ l(x_i, y_i) = 1\}$ can not be modified by the
      $SGT$ algorithm;
      $T[y_i] = SGT(S, T^*)$;
    end if
  end for
end if
return ($T$)
```

Note: Algorithm 1 and **Algorithm 2** describe the same Algorithm, **Algorithm 2** is a closer version to the implemented Algorithm.
Algorithm 2 as described leave aside the following important details actually implemented:

- The labelling of the T can be a float labelling from 0 to 1, T will then be "binary labelled" thanks to the threshold.
- The algorithm checks if the covering set of trees $\mathcal{T}$ is compatible with T, meaning that $\mathcal{T} = \{T[x_1], T[x_2], \ldots, T[x_n]\}$ such that $\cup_{i=1}^{n}\mathcal{L}(x_i) = \mathcal{L}(T)$ and $\cap_{i=1}^{n}\mathcal{L}(x_i) = \emptyset$
- The clades to be preserved are stored in a list, given to the *SGT* call. The *SGT* algorithm will consider these clades as leaves.
- On the first instance of *LabelGTC*, the clades to preserve included in $\mathcal{T}$ are added to the list of the clades to preserve (in order to respect *Section 6 : Accounting for the 0-1 edge labelling in partial subtrees*).
- After the modification of a subtree by a special case or the *SGT* call, this subtree is added to the clades to preserve.

▶ **Theorem 8.** *Algorithm 1 solves the* LabelGTC *problem on an instance* $\{S, T, \mathcal{T}, l\}$ *in time* $O(4^k.(n+1)^k.k)$ *where* $n = |\Gamma|$ *and* $k = |\mathcal{T}|$.

Heuristic algorithm for TripletGTC:

A natural heuristic algorithm for the TripletGTC problem on a gene tree T with a covering set $\mathcal{T}$ and a set of triplet Trp consists in first giving the label 1 to any edge (x, y) of $E(T) \setminus E(\mathcal{T})$ such that there exists a triplet (a, b, c) in Trp belonging to the set $Trp(x, y)$. Next the LabelGTC algorithm is applied to the obtained edge labelled tree. The corrected tree resulting from this algorithm will preserve all triplets of Trp, but more largely all triplets of the set $\{Trp(x, y) \mid (x, y) \in E(T) \setminus E(\mathcal{T})\ and\ l(x, y) = 1\}$, which includes Trp.

6 Accounting for the 0-1 edge labelling in partial subtrees

In the formulation of the LabelGTC problem, the 0-1 edge labels are ignored for the trees of $\mathcal{T}$, and only edges in $E(T) \setminus E(\mathcal{T})$ labelled 1 have to be preserved. A natural extension would be to preserve also the edges of $\mathcal{T}$ labelled 1.

This can be done by mean of Algorithm 1, but replacing the call to the SGT algorithm by an algorithm solving the following problem.

Label SuperGenetree (LabelSGT) Problem:
Input: A species tree S, a gene tree T, a covering set of trees $\mathcal{T}$ for T and a 0-1 edge labelling l for T.
Output: A supertree T' for $\mathcal{T}$ of minimum reconciliation cost such that: if $(x, y) \in E(\mathcal{T})$ is such that $l(x, y) = 1$, then there is an edge (x', y') in $E(T')$ such that $\mathcal{L}(y) = \mathcal{L}(y')$.

Before describing an algorithm solving the above LabelSGT problem, we first recall some useful definitions and the algorithm described in [7] for the SGT problem.

Given a gene tree T and a species tree S, $cost(T)$ denotes the reconciliation cost of T with S. If the root x of T has two children node, one of the subtree of T rooted at a child of x is arbitrarily denoted by T_l and the other one by T_r. Given a set of gene subtrees $\{t_1, \ldots, t_k\}$ and a bipartition (L_l, L_r) of $\bigcup_{i=1}^{k}\mathcal{L}(t_i)$, if T'_l and T'_r are two trees for L_l and L_r respectively,

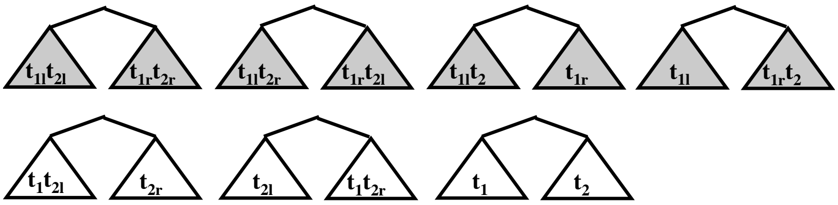

Figure 4 The 7 bipartitions of a set $\mathcal{B}(t_1, t_2)$. The first 4 bipartitions are those in which $\mathcal{L}(t_{1_l})$ and $\mathcal{L}(t_{1_r})$ are separated.

then (T_l, T_r) denotes the tree T such that $T_l = T'_l$ and $T_r = T'_r$. In this case, $cost(L_l, L_r)$ denotes the local reconciliation cost at the root x of the tree (T_l, T_r) counting the number of losses on the two edges linking x to T_l and T_l, plus 1 if x is a duplication node.

▶ **Definition 9** (Reformulation of Property 1 from [7])**.** Given a set of gene subtrees $\{t_1, \ldots, t_k\}$, $\mathcal{B}(t_1, \ldots, t_k)$ denotes the set of bipartitions (L_l, L_r) of $\bigcup_{i=1}^{k} \mathcal{L}(t_i)$ such that each subtree $t_i, 1 \leq i \leq k$, satisfies either: 1) $\mathcal{L}(t_i) \subseteq L_l$; or 2) $\mathcal{L}(t_i) \subseteq L_r$; or 3) $\mathcal{L}(t_{i_l}) \subseteq L_l$ and $\mathcal{L}(t_{i_r}) \subseteq L_r$; or 4) $\mathcal{L}(t_{i_l}) \subseteq L_r$ and $\mathcal{L}(t_{i_r}) \subseteq L_l$.

$\mathcal{B}(t_1, \ldots, t_k)$ contains exactly $\frac{4^k}{2} - 1$ bipartitions. For example, for $k = 2$ subtrees, the 7 bipartitions are depicted in Figure 4.

The following is the recurrence formulae of the dynamic programming algorithm described in [7] for the SGT problem.

▶ **Lemma 10** (Reformulation of Lemma 3 from [7])**.** *The following algorithm solves the* SGT *problem on an instance $\{S, T, \mathcal{T}\}$ such that $\mathcal{T} = \{t_1, \ldots, t_k\}$ in time $O(4^k.(n+1)^k.k)$ where $n = |\bigcup_{i=1}^{k} \mathcal{L}(t_i)|$.*

1. *(Stop condition) If $|\bigcup_{i=1}^{k} \mathcal{L}(t_i)| = 1$, then $SGT(t_1, \ldots, t_k)$ is the gene tree composed of the corresponding single node;*
2. *Otherwise, $SGT(t_1, \ldots, t_k) = (T'_l, T'_r)$ where $T'_l = SGT(t_{1|L_l}, \ldots, t_{k|L_l})$ and $T'_r = SGT(t_{1|L_r}, \ldots, t_{k|L_r})$ such that:*
$$(L_l, L_r) = \underset{(L_l, L_r) \in \mathcal{B}(t_1, \ldots, t_k)}{\text{argmin}} \{cost(L_l, L_r) + cost(T'_l) + cost(T'_r)\}$$

Let T be a 0-1 edge-labelled gene tree, and $\mathcal{T}$ be a covering set of subtrees for T. In order to preserve the edges in $\mathcal{T}$ labelled 1, it suffices to consider, at each step of the above algorithm, only the bipartitions that do not separate $\mathcal{L}(t_{i_l})$ and $\mathcal{L}(t_{i_r})$ for any subtree $t_i, 1 \leq i \leq k$ such that $t_i = T_j[y]$ and (x, y) is an edge of $E(T_j)$ labelled 1, unless $k = 1$. For example in Figure 4, if t_1 is such a subtree, then the four first bipartitions that separate $\mathcal{L}(T_{1_l})$ and $\mathcal{L}(T_{1_r})$ are discarded.

Given a set of gene subtrees $\{t_1, \ldots, t_k\}$ of $\mathcal{T}$, we define $\mathcal{B}_{Label}(t_1, \ldots, t_k)$ as the subset of $\mathcal{B}(t_1, \ldots, t_k)$ containing all bipartitions that do not separate any subtree $t_i, 1 \leq i \leq k$ such that $t_i = T_j[y]$, T_j is a tree of T and (x, y) is an edge of $E(T_j)$ labelled 1.

▶ **Theorem 11.** *The algorithm described in Lemma 10 in which any set $\mathcal{B}(t_1, \ldots, t_k)$ is replaced by the set $\mathcal{B}_{Label}(t_1, \ldots, t_k)$ solves the* LabelSGT *problem on $\mathcal{T}$ with the same time complexity as the initial algorithm, i.e. in time $O(4^k.(n+1)^k.k)$.*

7 Conclusion

This paper provides a unifying view allowing to reconcile apparently heterogeneous gene tree correction methods. The general LabelGTC and TripletGTC approaches have the advantage of not being dependent upon particular assumptions on trees. We present the first general algorithm allowing to correct a tree according to an arbitrary 0-1 edge-label, or more generally to an arbitrary set of triplets whose topology should be preserved. These algorithms have the same exponential time-complexity as the one previously developed for the SGT problem [7], which has been proved NP-hard for the duplication distance. Although no proof of complexity for the more general reconciliation distance exists, it is unlikely that adding losses makes the problems more tractable. We conjecture the SGT problem, and more generally the LabelGTC and TripletGTC problems, remain NP-hard in this case.

The 0-1 edge-labelling considered in this paper can be seen as a first step towards integrating knowledge on edge statistical support in a gene tree correction algorithm. Generalizing the edge-labelling function l to an arbitrary domain would require a complete reformulation of the problems. It may be more intuitive in this case to use a heuristic algorithm exploring a tree space around the input tree, and among statistically equivalent trees, take the one minimizing a combination of values accounting for both sequence alignment cost and reconciliation cost. Several such methods using species tree information in addition to sequence information, have been developed (e.g. TreeBeST [13], TreeFix [16], PhylDog [1], SPIMAP [11], ProfilNJ [10]). However, a formal conceptual framework, as the one developed in this paper, remains to be developed for the gene tree correction problem with a non-binary edge-labelling function.

References

1 B. Boussau, G.J. Szöllősi, L. Duret, M. Gouy, E. Tannier., and V. Daubin. Genome-scale coestimation of species and gene trees. *Genome Research*, 23:323-330, 2013.

2 K. Chen, D. Durand, and M. Farach-Colton. Notung: Dating gene duplications using gene family trees. *Journal of Computational Biology*, 7:429–447, 2000.

3 J. Felsenstein. Phylogenies from molecular sequences: Inference and reliability. *Ann. Review Genet.*, 22:521–565, 1988.

4 E. Jacox, C. Chauve, G.J. Szollosi, Y. Ponty, and C. Scornavacca. ecceTERA: comprehensive gene tree-species tree reconciliation using parsimony. *Bioinformatics*, 32(13):2056-2058, 2016.

5 M. Lafond, E. Noutahi, and N. El-Mabrouk. Efficient non-binary gene tree resolution with weighted reconciliation cost. In *Combinatorial Pattern Matching, LIPIcs-Leibniz International Proceedings in Informatics*, volume 54. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016.

6 M. Lafond, A. Ouangraoua, and N. El-Mabrouk. Reconstructing a supergenetree minimizing reconciliation. *BMC-Genomics*, 16:S4, 2015. Special issue of RECOMB-CG 2015.

7 N. El-Mabrouk M. Lafond, C. Chauve and A. Ouangraoua. Gene tree construction and correction using supertree and reconciliation. In *Asia Pacific Bioinformatics Conference*, 2017. soon in IEEE/ACM TCBB.

8 N. Nguyen, S. Mirarab, and T. Warnow. MRL and SuperFine+MRL: new supertree methods. *J. Algo. for Mol. Biol.*, 7(3), 2012.

9 Thi Hau Nguyen, Vincent Ranwez, Stéphanie Pointet, Anne-Muriel Arigon Chifolleau, Jean-Philippe Doyon, and Vincent Berry. Reconciliation and local gene tree rearrangement can be of mutual profit. *Algorithms Mol Biol*, 8(1):12, 2013. URL: `http://dx.doi.org/10.1186/1748-7188-8-12`, `doi:10.1186/1748-7188-8-12`.

10 E. Noutahi, M. Semeria, M. Lafond, J. Seguin, L. Gueguen, N. El-Mabrouk, and E. Tannier. Efficient gene tree correction guided by genome evolution. *Plos.One*, 11(8), 2016.

11 M.D. Rasmussen and M. Kellis. A bayesian approach for fast and accurate gene tree reconstruction. *Molecular Biology and Evolution*, 28(1):273- 290, 2011.

12 J.S.M. Sabir, R.K. Jansen, Sankoff $\cdots$ D, N. El-Mabrouk, and T.A. Ruhlman. The nuclear genome of Rhazya stricta and the evolution of alkaloid diversity in a medically relevant clade of apocynaceae. *Nature Scientific Reports*, 6(33782), 2016.

13 F. Schreiber, M. Patricio, M. Muffato, M. Pignatelli, and A. Bateman. Treefam v9: a new website, more species and orthology-on-the-fly. *Nucleic Acids Research*, 2013. doi: 10.1093/nar/gkt1055.

14 C. Scornavacca, L. van Iersel, S. Kelk, and D. Bryant. The agreement problem for unrooted phylogenetic trees is FPT. *Journal of Graph Algorithms and Applications*, 18(3):385 - 392, 2014.

15 A.J. Vilella, J. Severin, A. Ureta-Vidal, L. Heng, R. Durbin, and E. Birney. EnsemblCompara gene trees: Complete, duplication-aware phylogenetic trees in vertebrates. *Genome Research*, 19:327-335, 2009.

16 Y.C Wu, M.D. Rasmussen, M.S. Bansal, and M. Kellis. TreeFix: Statistically informed gene tree error correction using species trees. *Systematic Biology*, 62(1):110- 120, 2013.

17 Y. Zheng and L. Zhang. Reconciliation with non-binary gene trees revisited. In *Lecture Notes in Computer Science*, volume 8394, pages 418-432, 2014. Proceedings of RECOMB.

A Proofs

Proof of Theorem 4. Let T be a 0-1 edge-labelled gene tree, $\mathcal{T}$ be a covering set of subtrees for T and Trp be a set of triplets. If $Trp = \bigcup\{Trp(x, y) | (x, y) \in E(T) \setminus E(\mathcal{T}) and l(x, y) = 1\}$, then the TripletGTC problem on $\{T, \mathcal{T}, Trp\}$ is reduced to LabelGTC because a supertree T' of $\mathcal{T}$ preserves an edge (x, y) of $E(T) \setminus E(\mathcal{T})$ iff it preserves the topology of all triplets in $Trp(x, y)$. ◀

Proof of Theorem 6. The proof follows directly from the following three equalities:
(1) $Trp1 = \bigcup\{Trp(T_i) \mid T_i \in \mathcal{T}\}$;
(2) $Trp2 = \{Trp(x, y) \mid (x, y)\ is\ a\ terminal\ edge\ of\ T\}$;
(3) $Trp3 = \{Trp(x, y) \mid (x, y)\ is\ a\ non\ terminal\ edge\ of\ T\}$.

So a tree T' for $\mathcal{L}(T)$ preserves the topology of all trees $T_i \in \mathcal{T}$ iff it preserves $Trp1$; It preserves all terminal edges of $E(T)$ iff it preserves $Trp2$ and it preserves all non-terminal edges of $E(T)$ iff $Trp3$. The combination of the inclusion or exclusion of $Trp1$, $Trp2$ and $Trp3$ in Trp (except the case where $Trp = \emptyset$) results in the 7 cases of the theorem. ◀

Proof of Lemma 7. Let (x, y) be an edge in $E(T) \setminus E(\mathcal{T})$ such that $l(x, y) = 1$, and $\mathcal{E}$ be the largest covering set of edges of $E(T[y])$ that have no ancestral edges in $E(T[y])$ labelled 1. Suppose that $\mathcal{E}$ contains an edge (s, t) labelled 0 that is not a terminal edge and denote by t_l and t_r the two children of the node t. Then the edges (t, t_l) and (t, t_r) belong to $E(T) \setminus E(\mathcal{T})$ and satisfy the condition that they have no ancestral edges in $E(T[y])$ labelled 1. So $(\mathcal{E} \cup \{(t, t_l), (t, t_r)\}) \setminus \{(s, t)\}$ is also a covering set of edges of $E(T[y])$ that have no ancestral edges in $E(T[y])$ labelled 1, which contradicts the assumption that $\mathcal{E}$ is the largest such covering set of edges. ◀

Proof of Theorem 8. Let T be a 0-1 edge-labelled gene tree, $\mathcal{T}$ be a covering set of subtrees for T and $\mathcal{E} = \{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\}$ be the largest covering set of edges for T that have no ancestral edge labelled 1.

1. If all edges of $\mathcal{E}$ are labelled 0, then by Lemma 7, all the edges of $\mathcal{E}$ are terminal edges. Since $\mathcal{E}$ covers T, then all edges of $E(T) \setminus E(\mathcal{T})$ are labelled 0. So, by Theorem 1, the problem is reduced to SGT.
2. Otherwise, let $T' = LabelGTC(S, T, \mathcal{T}, l)$, and for any edge (x_i, y_i) of $\mathcal{E}$ labelled 1 and preserved in T', let y_i' be the node of T' such that $\mathcal{L}(y_i') = \mathcal{L}(y_i)$. Since $T_{|\mathcal{L}(y_i)}$ is a complete subtree of T, then $T'[y_i']$ is also a complete subtree of T' that can be constructed independently of the remaining of the tree T' as $T'[y_i'] = LabelGTC(S, T[y_i], \mathcal{T}_{|\mathcal{L}(y_i)}, l_{|E(T[y_i])})$ (computed at Step 2.a. of the algorithm). Next, the obtained subtrees must be adequately grafted on branches of a supergenetree of the remaining trees $\{T[y_i] \in \mathcal{T} \mid (x_i, y_i) \in \mathcal{E}\ and\ l(x_i, y_i) = 0\}$. In this case, the problem is also reduced to the SGT problem on $\mathcal{T}^* = \{T[y_i] : (x_i, y_i) \in \mathcal{E}\ and\ l(x_i, y_i) = 0\} \cup \{y_i^* : (x_i, y_i) \in \mathcal{E}\ and\ l(x_i, y_i) = 0\}$ where each tree $T'[y_i']$ computed at Step 2.a. is contracted into a single leaf node y_i^* associated to the leafset $\mathcal{L}(T'[y_i'])$ (computed at Steps 2.b. to 2.e. of the algorithm).

The worst case of the algorithm is the stop case where it is directly reduced to the SGT algorithm whose time complexity is in $O(4^k.(n + 1)^k.k)$. ◀

Proof of Theorem 11. The algorithm described in Lemma 10 solves the SGT problem by exploring the set of all possible supergenetrees for $\mathcal{T}$. The set of supergenetrees is explored by considering all possible bipartitions (L_l, L_r) for each set $\bigcup_{i=1}^{k} \mathcal{L}(t_i)$ where each t_i is a subtree of a tree of $\mathcal{T}$. So, replacing any set $\mathcal{B}(t_1, \ldots, t_k)$ by the set $\mathcal{B}_{Label}(t_1, \ldots, t_k)$ will only discard the bipartitions (L_l, L_r) that prevent the preservation of an edge of $E(\mathcal{T})$ labelled 1. So the modification of the algorithm will return a supergenetree for $\mathcal{T}$ of minimum reconciliation cost preserving all edges of $E(\mathcal{T})$ labelled 1. The time complexity of the modified algorithm remains the same as that of the initial algorithm. ◀