Reconciling a Gene Tree to a Species Tree Under the Duplication Cost Model*

Paola Bonizzoni† Gianluca Della Vedova‡ Riccardo Dondi§

April 12, 2005

Abstract

The general problem of reconciling the information from evolutionary trees representing the relationships between distinct gene families is of great importance in Bioinformatics and has been popularized among the Computer Science researchers in [7] (Ma, Li and Zhang, SIAM J. Comp. 2000) where the authors pose the intriguing question if a certain definition of minimum tree that reconciles a gene tree and a species tree is correct. We answer affirmatively to this question; moreover we show an efficient algorithm for computing such minimum-leaf reconciliation trees and prove the uniqueness of such trees. We then tackle some different versions of the biological problem by showing that the exemplar problem, arising from the exemplar analysis of multigene genomes, is **NP**-hard even when the number of copies of a given label is at most two. Finally we introduce two novel formulations for the problem of recombining evolutionary trees, extending the gene duplication problem studied in [7, 3, 8, 9, 6], and we give an exact algorithm (via dynamic programming) for one of these formulations.

1 Introduction

Reconstructing and comparing evolutionary trees are two topics of Computational Biology that pose several new challenging problems to the Computer Science community. An evolutionary tree or *species tree* is a rooted binary tree where each leaf is labeled by a taxon in a set of extant species; the tree represents the ancestor-descendant relationships between species in the tree. Species trees are related to gene trees. These are also rooted binary trees differing from species trees in the way leaves of the tree are labeled. The labels of a gene tree G and a species tree S over the same set of species are taken from the same set L of labels, but any given label occurs at most once in S.

A widely used strategy for constructing a species tree consists of two basic steps. Given a gene (or a gene family) for the extant species, the first step constructs a gene tree representing the relationships among the sequences encoding that gene in the different species. Indeed, a

*A preliminary version of the results of this paper has appeared in the Proceedings of the 5th Italian Conference on Algorithms and Complexity

†Dipartimento di Informatica, Sistemistica e Comunicazione, Università degli Studi di Milano-Bicocca Milano - Italy, bonizzoni@disco.unimib.it

‡Dipartimento di Statistica, Università degli Studi di Milano-Bicocca Milano - Italy, gianluca.dellavedova@unimib.it

§Dipartimento di Informatica, Sistemistica e Comunicazione, Università degli Studi di Milano-Bicocca Milano - Italy, riccardo.dondi@unimib.it

gene family is represented by homologous sequences and the initial assumption is that such genes evolve in the same way as the species. The second step consists of deriving or inferring the species tree from the gene tree, where a species tree represents unambiguously the relations among species, as gene trees can differ from the actual species tree because of some typical biological phenomena related to evolutionary events, such as gene divergence resulting from duplications, losses and other gene mutations. Duplications are common evolutionary events and consist of copying in multiple places a gene located along a DNA strand. Then all those copies evolve independently from each other. In a species tree the fact that an internal node has two children represents a speciation event in which a species has evolved into two different species. In a gene tree an internal node with two children represents either a speciation event or the effect of an evolutionary event localized in a single gene.

Because of duplications, not only may gene trees for different gene families not agree, but even a single gene tree may not be completely consistent with any species tree; consequently the need for summarizing often contradictory gene trees into a unique species tree arises. A recent and biologically successful approach to this problem is the gene duplication model, proposed by various authors [5, 8, 6], which is based on introducing a mapping M from nodes of a gene tree G to nodes of a species tree S. In a gene tree, an internal node g represents an ancestral gene which is associated by M with the most recent ancestral species of S (an internal node of tree S) that contains all contemporary genes (leaves of the gene tree) descending from g. The mapping M is computationally modeled by using the *least common ancestor* (lca) mapping in a species tree.

Two main problems have emerged in the study of gene trees and species trees: the comparison of gene trees and species trees, in order to construct a tree that reconciles the evolutionary history represented by the two trees, and the inference of a species tree from a given gene tree.

A problem, pointed out in [5, 8] as a valid biological means to study gene phylogenies, is that of finding the best tree that reconciles a gene tree to a species tree, more precisely in [8] a notion of reconciled tree as a smallest tree satisfying three basic properties has been defined (see Def. 3.1 for the formal definition of those properties).

In [7] a recursive definition of reconciled tree is used to give an algorithm to construct such a tree. In the same paper the authors posed the question whether their construction actually produces a smallest reconciled tree, thus leaving open the problem of giving an effective algorithm to construct a smallest reconciled tree. In our paper we solve the open question mentioned above. Indeed, we propose a recursive algorithm to build *the minimum reconciled tree*: our construction allows us also to show that such a tree is unique.

In [7] the reconstruction of a species tree from a gene tree, under the duplication cost model is proved **NP**-hard when the number of occurrences of a label (i.e. a gene copy) in the gene tree is unbounded, pointing out that having multiple copies of a gene makes the problem difficult (or impossible) to solve efficiently. This motivates the search for some alternative sound formulations of the reconciliation problem. In this paper we will investigate an approach based on the notion of exemplar. This notion is already used in genome rearrangement to deal with multigene families when comparing two genomes; the idea is to keep only one of the multiple identical copies, so that some rearrangement distance is minimized. In this framework we define an exemplar tree as a tree obtained from the gene tree by keeping only one occurrence of each distinct gene while minimizing the total number of duplications. We show that even if a gene occurs in at most two copies, the exemplar tree problem is **NP**-hard.

The third and final part of our paper introduces some variants of the original recombination

problem of [7]. Initially we define and solve efficiently (via dynamic programming) a new problem in which the gene tree G and the species tree S are given and we look for a minimum cost mapping from the nodes of G to those of S. We generalize the reconciliation problem to that of finding a mapping and a species tree of minimum cost. Finally we present some possible directions of future research.

2 Gene trees and species trees

In this section we will present some basic terminology used in the paper, following the notation used in [7] to introduce the fundamental problem of reconciling gene trees to species trees.

In the paper we focus on evolutionary trees. For this reason, unless otherwise stated, all trees are *binary*, that is each internal node x of a tree has exactly two children, the left one denoted by $a(x)$ and the right one denoted by $b(x)$. Moreover all trees T are *rooted*, such that there is a distinguished vertex, called *root* and denoted by $r(T)$, that has no ancestor. Given a tree T we will denote by T_a (T_b) the subtree of T rooted at $a(r(T))$ ($b(r(T))$).

We consider two different kinds of evolutionary trees: species trees and gene trees, where species trees represent a restricted case of gene trees. Gene trees have labeled leaves, while the internal nodes are unlabeled. Given a gene tree T, we denote by $\Lambda(T)$ its leafset and by $L(T)$ the set of labels of its leaves. Given a node x of a tree T, the *cluster* of x, denoted by $\mathcal{C}(x)$, is the set of labels of all leaves of T that are descendants of x. Moreover, $\mathcal{C}(T)$ denotes the set of clusters of all internal nodes of T. In the following, in order to simplify the notation we choose to denote a cluster associated to a node x by the node itself. A *species* tree S is a gene tree whose leafset is *uniquely labeled* (no two leaves share a common label); consequently the sets $\Lambda(S)$ and $L(S)$ are isomorphic.

In the following we will always use G to denote a gene tree, S to denote a species tree and T for a generic tree. Given a pair (G, S), the *lca mapping*, in short lca, is a function that associates to each node g of G the node s of S such that $\mathcal{C}(s)$ is the smallest cluster of S containing $\mathcal{C}(g)$. Please note that the lca function is unique for any given pair (G, S).

The notion of *homomorphic subtree* is central in our paper. Given a labeled tree T and a subset L_1 of the leaves of T, the homomorphic subtree T_1 of T induced by L_1 is obtained by first removing all nodes and edges of T that are not in a path from the root of T to a leaf in L_1, and then contracting each internal node x that has only one child, that is creating an edge connecting the two neighbors of x in T, and finally removing x and all edges incident on it.

3 The reconciled tree problem

A gene tree G and a species tree S are comparable if $L(G) \subseteq L(S)$. Given a pair (G, S) of comparable trees a basic approach, proposed in [8] and used in [7], for gene tree reconciliation consists of computing a *reconciled tree* $R(G, S)$. The general notion of reconciled tree is that of a tree which contains G as a homomorphic subtree and also represents the evolutionary history of S by having exactly the same clusters as the species tree S. Clearly, for any pair (G, S) there exists an infinite number of reconciled trees but, according to the principle of maximum parsimony, we are interested only in the smallest one, that is the tree with the minimum number of leaves: we will denote such tree as *minimum* reconciled tree $R(G, S)$, or simply R whenever it

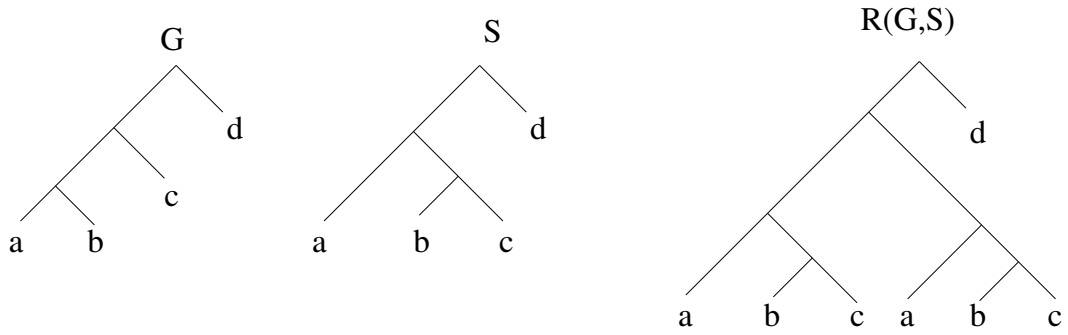

Figure 1: A gene tree G, a species tree S and a reconciled tree for (G, S)

is not ambiguous. In [8] a formal definition of a minimum reconciled tree has been introduced; below is the version of this definition given in [7].

Definition 3.1. A *minimum reconciled tree* $R(G, S)$ of a gene tree G and a species tree S, or in short for a pair (G, S), is a smallest tree satisfying the following properties:

1. G is a homomorphic subtree of $R(G, S)$,
2. $\mathcal{C}(R(G, S)) = \mathcal{C}(S)$,
3. for any internal node x of $R(G, S)$ either $\mathcal{C}(a(x)) \cap \mathcal{C}(b(x)) = \emptyset$ or $\mathcal{C}(a(x)) = \mathcal{C}(b(x)) = \mathcal{C}(x)$.

In [7] a recursive definition of a general reconciled tree has been introduced in order to describe an efficient algorithm for constructing such tree: this definition is reported here as Def. 3.2. The authors of [7] ask whether their definition and Def. 3.1 are equivalent. Indeed, they could not prove that their construction is optimal w.r.t. the size of the reconciled tree. We will answer affirmatively to this question by giving a simpler but equivalent recursive definition of a minimum reconciled tree (Def. 3.3); moreover we can prove that the optimal (smallest) tree is unique. It is immediate to note that the minimum reconciliation tree of the empty gene tree (that is a gene tree with no leaves) and a species tree S is isomorphic to S. We will denote the empty gene tree with $\emptyset$.

In the following we introduce several operations on trees proposed in [7]: *restriction*, *composition* and *replacement*. In the definition of the operations let T_1, T_2 be two trees and let t be a node of T_1.

- The *restriction* of T_1 at t, denoted by $T_1|t$, is the subtree of T_1 rooted at t.
- The *composition* of T_1, T_2, denoted by $T_1 \triangle T_2$, consists of the (rooted binary) tree T such that $T_a = T_1$ and $T_b = T_2$. Informally T is obtained by adding a node r and connecting r to the roots of the two trees T_1 and T_2.
- The *replacement of* T_1 *with* T_2 at t, denoted by $T_1(t \to T_2)$, is the tree obtained by replacing in T_1 the subtree rooted at t with T_2.

In particular, the replacement operation can be generalized to two nodes t, t' in a tree T_1 as follows: then $T_1(t \to T_2, t' \to T_3)$ is the tree obtained by replacing in T_1 the subtrees rooted at t, t' with T_2, T_3, respectively.

The composition and replacement operation are then used to give the two recursive definitions of reconciled tree.

Definition 3.2. [7] Let G, S be respectively a gene tree and a species tree, then $R(G,S)$ is equal to G if G and S consist of a single leaf, otherwise $R(G,S)$ is equal to:

1. $S(a(r(S)) \rightarrow R(G, S_a))$ if $\text{lca}(r(G)) \subseteq a(r(S))$
2. $S(\text{lca}(a(r(G))) \rightarrow R(G_a, S|\text{lca}(a(r(G)))), \text{lca}(b(r(G))) \rightarrow R(G_b, S|\text{lca}(b(r(G)))))$, if $\text{lca}(r(G)) = r(S)$ and $\text{lca}(a(r(G)))$ and $\text{lca}(b(r(G)))$ are mapped to $s_1 \subseteq a(r(S))$ and $s_2 \subseteq b(r(S))$ respectively,
3. $S(\text{lca}(a(r(G))) \rightarrow R(G_a, S|\text{lca}(a(r(G))))) \triangle R(G_b, S)$, if $\text{lca}(r(G)) = \text{lca}(b(r(G))) = r(S)$ and $\text{lca}(a(r(G))) \subseteq a(r(S))$.
4. $R(G_a, S) \triangle R(G_b, S)$ if $\text{lca}(r(G))$, $\text{lca}(a(r(G)))$, $\text{lca}(b(r(G)))$ are all equal to $r(S)$.

Definition 3.3. Let G, S be respectively a gene tree and a species tree, then $R(G,S)$ is equal to G if G and S are both leaves, otherwise $R(G,S)$ is equal to:

1. $S(\text{lca}(a(r(S))) \rightarrow R(G, S_a)))$ if $\text{lca}(r(G)) \subseteq a(r(S))$.
2. $R(G_a, S_a) \triangle R(G_b, S_b)$, if $\text{lca}(r(G)) = r(S)$, $\text{lca}(a(r(G)))$ and $\text{lca}(b(r(G)))$ are mapped to $s_1 \subseteq a(r(S))$ and $s_2 \subseteq b(r(S))$ respectively
3. $R(G_a, S) \triangle R(G_b, S)$, if $\text{lca}(r(G)) = r(S)$, and at least one of $\text{lca}(a(r(G)))$ and $\text{lca}(b(r(G)))$ is equal to $r(S)$.

For Def. 3.2 and Def. 3.3 the case $\text{lca}(r(G)) \subseteq b(r(S))$ is symmetric to $\text{lca}(r(G)) \subseteq a(r(S))$ in point (1) of each of these definitions. Notice that Def. 3.3 leads naturally to a recursive algorithm where each point of the definition modifies the two trees G and S to be reconciled so that $L(G) \subseteq L(S)$ always holds; moreover the termination condition is reached when G and S are both leaves.

4 Proving the conjecture

Note that the conditions stated in Def. 3.3 cover all possible cases that occur when the lca-mapping associates the root of G and its children to nodes of S; moreover those conditions are mutually exclusive. Hence Def. 3.3 is well formed. More precisely, each point of Def. 3.2, 3.3 explains how the clusters of the two topmost levels of G relate to those of S under certain conditions. Such conditions are listed below and are illustrated in Fig. 4:

1a The root of G is mapped to one of the descendants of $r(S)$. This is considered in point 1 of Def. 3.2 and Def. 3.3. Note that these points are identical.

1b The root of G, $r(G)$, is mapped to $r(S)$ and its children are mapped to the subtrees S_a, S_b respectively. This case is considered in point 2 of Def. 3.2 and Def. 3.3.

1c The root of G, $r(G)$, is mapped to $r(S)$ and at least one child of $r(G)$ is mapped to $r(S)$. This case is considered in point 3 of Def. 3.2 and in points 3 and 4 of Def. 3.3.

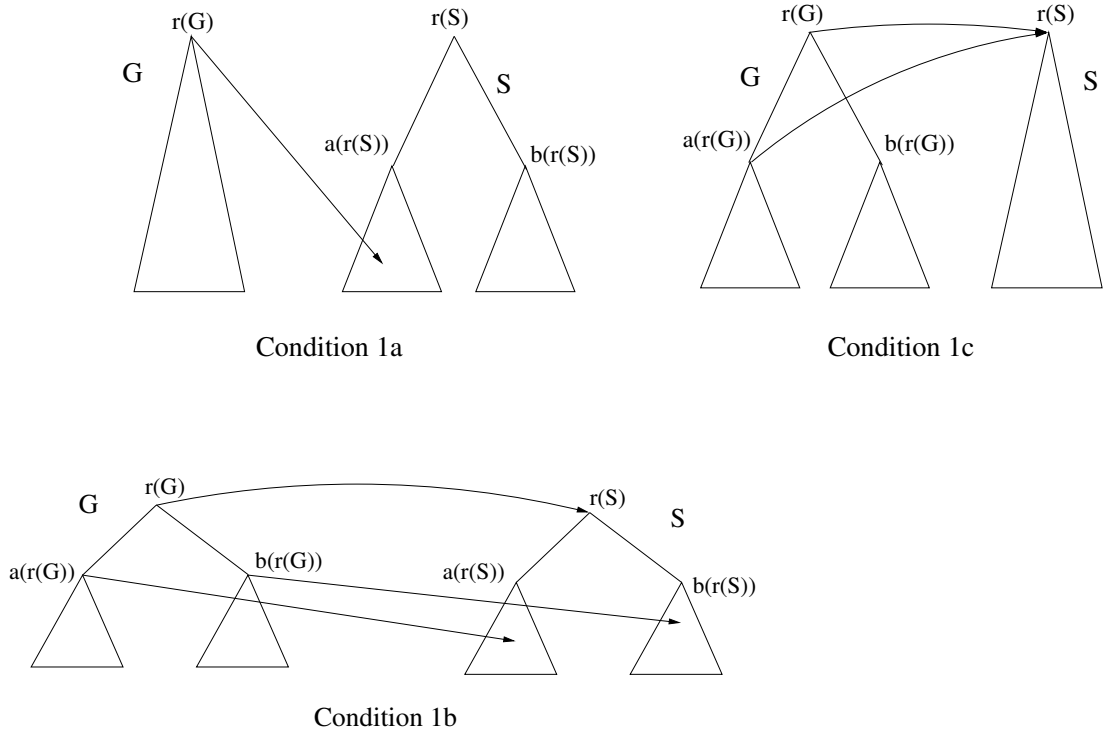

Figure 2: Conditions 1a-1c

As above, given a reconciled tree $R(G, S)$ we will study the following three mutually exclusive conditions that cover all possible relations between the subtree of R homomorphic to G and the two topmost levels of G and S:

2a There exists a subtree T in R contained in $a(R)$ that is homomorphic to G.

2b all the subtrees T of R homomorphic to G have root $r(R)$, and $\mathcal{C}(a(r(R))) \cap \mathcal{C}(b(r(R))) = \emptyset$.

2c all the subtrees T of R homomorphic to G have root $r(R)$, and $\mathcal{C}(a(r(R))) = \mathcal{C}(b(r(R)))$.

We will call conditions of *type 1* those enumerated as 1a,1b and 1c and conditions of *type 2*, those enumerated as 2a,2b and 2c. We will prove that *under the assumption that $R(G, S)$ is a minimum reconciled tree*, then condition 1a is equivalent to condition 2a, 1b is equivalent to 2b and 1c is equivalent to 2c, by showing that condition 2a implies 1a (Prop. 4.5), 2b implies 1b (Prop. 4.4), and 2c implies 1c (Prop. 4.3). Those implications suffice to prove the equivalence of each condition of type 1 to the condition of type 2 with same label, as indeed conditions 2a, 2b, 2c cover all possibilities. The following observation will be used in some of the following proofs.

Observation 4.1. *Let R be a reconciled tree for (G, S), and let x be an internal node of R. Then there exists an internal node y of S such that $\mathcal{C}(R|x) = \mathcal{C}(S|y)$. Moreover if $\mathcal{C}(a(R|x)) \neq \mathcal{C}(b(R|x))$ then $\mathcal{C}(a(R|x)) = \mathcal{C}(a(S|y))$ and $\mathcal{C}(b(R|x)) = \mathcal{C}(b(S|y))$.*

We leave to the reader the proof that the tree constructed according to Def. 3.3 satisfies the three properties of Def. 3.1. It is not clear whether such a tree is a smallest tree with such properties and, if this case holds, whether this tree is unique. A classical result from graph theory (Remark 4.1) leads to Lemma 4.2.

Remark 4.1. *Let T_1 and T_2 be two uniquely labeled binary trees such that $\mathcal{C}(T_1) = \mathcal{C}(T_2)$. Then T_1 is isomorphic to T_2.*

We recall that two clusters of a tree T are either disjoint or one is contained in the other; moreover any cluster which is not a leaf is always a union of two clusters.

Lemma 4.2. *Let T be a uniquely labeled binary tree and let T' be a binary tree (not necessarily uniquely labeled) such that $L(T) \subseteq L(T')$ and $\mathcal{C}(T) \supseteq \mathcal{C}(T')$. Then $L(T) = L(T')$ and $\mathcal{C}(T) = \mathcal{C}(T')$.*

Proof. Assume that $L(T) \subset L(T')$ and $l \in L(T') - L(T)$. Then $\{l\}$ is a cluster of T' but not of T, contradicting the assumption that $\mathcal{C}(T) \supseteq \mathcal{C}(T')$.

We can now assume that $L(T) = L(T')$. Suppose $\mathcal{C}(T) \supset \mathcal{C}(T')$ and let c_1 be a minimum cluster such that $c_1 \in \mathcal{C}(T) - \mathcal{C}(T')$. Since $L(T) = L(T')$, then $|c_1| \geq 2$. Thus let c_a, c_b be the two clusters of T that are children of c_1 in T, that is $c_1 = c_a \cup c_b$. By minimality of c_1 it follows that both c_a, $c_b \in \mathcal{C}(T')$. Let c_1^* be the minimum cluster of T' such that $c_1^* \supset c_1$.

Now let us consider the occurrence of c_1^* in T' that is furthest from the root, and let c_a^* and c_b^* be the two children of such occurrence in T'. Clearly c_a^* and c_b^* are two nodes of T whose union is c_1^*, consequently c_a^* and c_b^* are disjoint or one included in the other one. If c_a^* and c_b^* are disjoint then they must be siblings in T, hence they are the children of c_1^* in T, contradicting the hypothesis that c_1^* is the smallest cluster of T' properly including c_1. Conversely if c_a^* includes c_b^* and their union is c_1^*, c_a^* is an occurrence of c_1^* that is further from the root than the one chosen originally, which is a contradiction. □

Proposition 4.3. *Let R be a minimum reconciled tree for (G, S) such that $r(R)$ is the root of all the subtrees T of R homomorphic to G and $\mathcal{C}(R_a) = \mathcal{C}(R_b)$. Then at least one of $lca(a(r(G)))$ and $lca(b(r(G)))$ is equal to $r(S)$.*

Proof. First we prove that $r(G)$ is mapped in $r(S)$. Since $L(G) \subseteq L(S)$, $\mathrm{lca}(r(G)) \subseteq r(S)$. Suppose now that $\mathrm{lca}(r(G)) \subseteq a(r(S))$, consequently there exists a label in $L(S)$ that is not in $L(G)$: let l be one such label. Since $\mathcal{C}(R_a) = \mathcal{C}(R_b)$ a copy of l must be in both $R_a = R_b$, hence l is the label of at least two leaves of R. Let T be the homomorphic subtree of R obtained removing all leaves labeled l, consequently $|L(R) - L(T)| \geq 2$. Clearly $T \triangle \{l\}$ is a reconciliation tree for (G, S) with less leaves than R, which is a contradiction with the minimality of R, thus proving that $\mathrm{lca}(r(G)) = r(S)$.

Now we have to prove that $\mathrm{lca}(a(r(G))) = r(S)$ or $\mathrm{lca}(b(r(G))) = r(S)$. Assume to the contrary that $\mathrm{lca}(a(r(G))) \subseteq a(r(S))$ and $\mathrm{lca}(b(r(G))) \subseteq b(r(S))$. Now let R'_a (respectively R'_b) be the tree obtained from R_a (resp. R_b) removing all leaves whose labels are not in S_a, (resp. R_b).Clearly, such labels exist as $\mathcal{C}(S_a) \subset \mathcal{C}(R_a)$. Since $r(R)$ is also the root of the subtree T homomorphic to G, the tree $R_a \triangle R_b$ contains T. The second and third points of Def. 3.1 trivially hold for $R_a \triangle R_b$, which therefore is a reconciled tree for (G, S). By construction $R_a \triangle R_b$ is smaller than R which is again a contradiction with the minimality of R. □

Proposition 4.4. *Let R be a minimum reconciled tree for (G, S) such that $r(R)$ is the root of all the subtrees T of R homomorphic to G and $\mathcal{C}(R_a) \cap \mathcal{C}(R_b) = \emptyset$. Then $lca(r(G)) = r(S)$, $lca(a(r(G))) \subseteq a(r(S))$ and $lca(b(r(G))) \subseteq b(r(S))$.*

Proof. Since $\mathcal{C}(R) = \mathcal{C}(S)$ and $\mathcal{C}(R_a) \cap \mathcal{C}(R_b) = \emptyset$, it is not restrictive to assume that $\mathcal{C}(R_a) = \mathcal{C}(S_a)$ and $\mathcal{C}(R_b) = \mathcal{C}(S_b)$. Since by hypothesis $r(R)$ is the root of all the subtrees T of R homomorphic to G, it follows that T_a is in R_a and thus all leaves of $a(T)$ are in $L(a(R))$. Consequently $\text{lca}(a(r(G))) \subseteq a(r(R))$ and thus $\text{lca}(a(r(G))) \subseteq a(r(S))$, by definition of least common ancestor. An analogous argument shows that $\text{lca}(b(r(G))) \subseteq b(r(S))$.

Now we will prove that $\text{lca}(r(G)) = r(S)$.Since $L(G) \subseteq L(S)$, it holds that $\text{lca}(r(G)) \subseteq r(S)$. Assume to the contrary that $\text{lca}(r(G)) \subseteq a(r(S))$ (a similar argument holds when considering the right child of $r(S)$). Then $L(G) \subseteq S_a$ and, since $\mathcal{C}(R_a) = \mathcal{C}(S_a)$ as shown above, it follows that $L(G) \subseteq R_a$. Since $\mathcal{C}(R_a)$ and $\mathcal{C}(R_b)$ are disjoint, $\mathcal{C}(S_a)$ and $\mathcal{C}(S_b)$ must also be disjoint; therefore no label of $L(G)$ is in $L(S_b)$ (nor it is in R_b). Now, for the properties of homomorphic subtree, there exists a subtree of R having root in $a(r(R))$ and homomorphic to G thus contradicting the original assumption that every homomorphic subtree of G have root in R.

□

Proposition 4.5. *Let R be a minimum reconciled tree for (G, S), such that there exists a subtree T in R_a homomorphic to G. Then lca$(r(G)) \subseteq a(r(S))$.*

Proof. Notice that $\mathcal{C}(R_a) \cap \mathcal{C}(R_b) = \emptyset$. In fact assume to the contrary that $\mathcal{C}(a(r(R))) = \mathcal{C}(b(r(R))) = \mathcal{C}(r(R)) = \mathcal{C}(r(S))$, where the last two equalities descend directly from points 2,3 of Def. 3.1. Thus R_a is a reconciled tree for (G, S), since it contains a homomorphic copy of G and contains all the clusters of S, which contradicts the minimality of R.

Since $\mathcal{C}(a(r(R)))$ and $\mathcal{C}(b(r(R)))$ are disjoint, $\mathcal{C}(a(r(R))) = \mathcal{C}(a(r(S)))$, by definition 3.1 of reconciled tree. But the leaves of T are included in the leaves of $a(R)$, and consequently in those of $a(S)$ which, in turn, implies that $\text{lca}(r(G)) \subseteq a(r(S))$ by definition of least common ancestor. □

Corollary 4.6. *Let R be a minimum reconciliation tree for (G, S). Then conditions 1a, 1b, 1c are respectively equivalent to conditions 2a, 2b, 2c.*

Now that the equivalence of conditions 1a-1c and 2a-2c has been established, with the following three lemmas we will show that a minimum reconciliation tree for (G, S) must satisfy one of conditions 2a, 2b, 2c.

Lemma 4.7. *Let R be any minimum reconciled tree for (G, S) such that there exists a subtree T of R homomorphic to G having the root in R_a. Then R_a and R_b must be respectively a minimum reconciliation tree for (G, S_a) and for $(\emptyset, S_b)$.*

Proof. From the proof of Prop. 4.5 we already know that $\mathcal{C}(R_a) \cap \mathcal{C}(R_b) = \emptyset$, hence $\mathcal{C}(a(r(R))) = \mathcal{C}(a(r(S)))$, by Lemma 4.2. Consequently, since R_a contains T, R_a must be a reconciled tree for (G, S_a). Assume that R_a is not minimum, and let R' be a minimum reconciled tree for (G, S_a). Then it is immediate to note that $R(a(r(R)) \rightarrow R')$ is a reconciliation tree for (G, S), contradicting the original assumption that R is a minimum reconciliation tree for (G, S). Since $\mathcal{C}(R_a) \cap \mathcal{C}(R_b) = \emptyset$, the clusters of R_b are exactly those of S_b, consequently R_b is a reconciliation tree for $(\emptyset, S_b)$. Similar reasoning establishes that R_b is a minimum reconciliation tree for $(\emptyset, S_b)$. □

Lemma 4.8. *Let R be any minimum reconciled tree for (G, S) such that $r(R)$ is the root of all the subtrees T of R homomorphic to G and $\mathcal{C}(R_a) \cap \mathcal{C}(R_b) = \emptyset$. Then R_a and R_b are respectively a minimum reconciled tree for (G_a, S_a) and (G_b, S_b).*

Proof. Since $\mathcal{C}(R_a) \cap \mathcal{C}(R_b) = \emptyset$, by definition 3.1, condition 2, it holds that $\mathcal{C}(R_a) = \mathcal{C}(S_a)$. Moreover since all nodes of T_a are also in G_a, R_a is a reconciled tree for (G_a, S_a). Assume that R_a is not minimum, and let R' be a minimum reconciled tree for (G_a, S_a). Then it is immediate to note that $R(R_a \rightarrow R')$ is a reconciliation tree for (G, S), contradicting the original assumption that R is a minimum reconciliation tree for (G, S).

An analogous argument can be used to prove that R_b is a minimum reconciled tree for (G_b, S_b). □

Lemma 4.9. *Let R be any minimum reconciled tree for (G, S) such that $r(R)$ is the root of all the subtrees T of R homomorphic to G and $\mathcal{C}(a(r(R))) = \mathcal{C}(b(r(R)))$. Then R_a and R_b are respectively a minimum reconciled tree for (G_a, S) and (G_b, S).*

Proof. We will prove that R_a must be a reconciled tree for (G_a, S). In fact R_a contains a homomorphic copy of $T|a(r(T))$, since $r(R)$ is the root of T and thus condition 1 of definition 3.1 holds. Moreover $\mathcal{C}(R_a) = \mathcal{C}(S)$ and thus conditions 2, 3 of the same definition 3.1 hold. Assume that R_a is not minimum, and let R' be a minimum reconciled tree for (G_a, S). Then it is immediate to note that $R(R_a \rightarrow R')$ is a reconciliation tree for (G, S) which is smaller than R, contradicting the original assumption that R is a minimum reconciliation tree for (G, S). The same reasoning establishes that R_b contains a homomorphic copy of $T|b(r(T))$ □

Lemma 4.10. *Let R be a minimum reconciled tree for (G, S). Then R is constructed according to Def. 3.3.*

Proof. We prove the lemma by induction on the number k of leaves in the tree R. If $k = 1$ then G and S are single leaves and $G = S$, hence R is constructed as in Def. 3.3.

Assume now that R is a minimum reconciled tree on k leaves and notice that minimum reconciled tree on less than k leaves is constructed as in Def. 3.3. Then R must verify exactly one of conditions 2a, 2b, 2c, being conditions 2a, 2b and 2c mutually exclusive. For each condition of type 2, one of Lemmas 4.7, 4.8 or 4.9 applies. Thus the following cases must be considered.

Case 1. Assume that condition 2a holds. By Lemma 4.7, then R_a and R_b are respectively a minimum reconciled tree for (G, S_a) and for $(\emptyset, S_b)$. By inductive hypothesis, tree R_a and R_b are constructed by Def. 3.3. By Corollary 4.6 condition 1a holds, that is the root of G is mapped to a descendent of $r(S)$. Then, it is immediate to verify that R is constructed as required in point 1 of Def. 3.3.

Case 2. Assume that condition 2b holds. By Lemma 4.8, then R_a and R_b are minimum reconciled trees for (G_a, S_a) and (G_b, S_b). Thus, by inductive hypothesis R_a, R_b are constructed from Def. 3.3. By Corollary 4.6, condition 2b is equivalent to condition 1b, which is the one required for the application point 2 of Def. 3.3 and consequently it is immediate to verify that R is constructed according to the operation of Def. 3.3.

Case 3. Assume that condition 2c holds. By Lemma 4.9, then R_a and R_b are minimum reconciled trees for (G_a, S) and (G_b, S). Thus, R_a, R_b are constructed from Def. 3.3. Moreover, by Corollary 4.6, condition 2c is equivalent to condition 1c, which is the one required for the application of point 3 of Def. 3.3. Again it can be verified that R is constructed from point 3 of Def. 3.3. □

Theorem 4.11. *Let R be a reconciled tree for (G, S) constructed according to Def. 3.3. Then R is unique.*

Proof. We prove the uniqueness of R by induction on the number k of leaves in the tree R constructed as in Def. 3.3. If $k = 1$ then G and S are single leaves and $G = S$. The computed tree is exactly S and it is unique.

Assume now that $k > 1$ and observe that R must have at least an internal node. Moreover, for inductive hypothesis, assume that the recursive definition Def. 3.3 gives a unique reconciled tree R when the number of leaves is smaller than k.

Now let R be a reconciled tree of (G, S). Thus R must satisfy exactly one of conditions of type $1a - 1c$, since they are mutually exclusive. Moreover, this condition is uniquely determined by G and S, i.e. it is the same for all reconciled trees of G and S.

Case 1. In this case condition 1a holds and every reconciled tree R constructed by Def. 3.3 is such that R_a is a reconciled tree of G and S_a and R_b is a reconciled tree of a gene tree with empty leafset and S_b. By inductive hypothesis R_a and R_b are unique, thus implying also that R is unique.

Case 2. In this case condition 1b holds and every reconciled tree R constructed by Def. 3.3 is such that R_a is a reconciled tree of (G_a, S_a) and and R_b is a reconciled tree of (G_b, S_b). By inductive hypothesis R_a and R_b are unique, thus implying also that R is unique.

Case 3. In this case condition 1c holds and every reconciled tree R constructed by Def. 3.3 is such that R_a is a reconciled tree of (G_a, S) and and R_b is a reconciled tree of (G_b, S). By inductive hypothesis R_a and R_b are unique, thus implying also that R is unique.

□

Corollary 4.12. *Given a gene tree G and species tree S the reconciled tree is unique.*

Proof. Given (G, S) and let R be a minimum reconciled tree of (G, S). Hence R must be constructed following Def. 3.3 and thus, by Lemma 4.11, R is unique. □

We conclude this section by proving that Definitions 3.2 and 3.3 are equivalent. Before proving the equivalence of Definitions 3.2 and 3.3, we restate Def. 3.3, splitting point 3 of Def. 3.3.

Definition 4.1. Let G, S be respectively a gene tree and a species tree, then $R(G, S)$ is equal to G if G and S are both leaves, otherwise $R(G, S)$ is equal to:

1. $S(S_a \to R(G, S_a))$ if $\text{lca}(r(G)) \subseteq a(r(S))$.
2. $R(G_a, S_a) \triangle R(G_b, S_b)$, if $\text{lca}(r(G)) = r(S)$, $\text{lca}(a(r(G)))$ and $\text{lca}(b(r(G)))$ are mapped to $s_1 \subseteq a(r(S))$ and $s_2 \subseteq b(r(S))$ respectively
3. $R(G_a, S) \triangle R(G_b, S)$, if $\text{lca}(r(G)) = \text{lca}(b(r(G))) = r(S)$, and $\text{lca}(a(r(G))) \subseteq a(r(S))$.
4. $R(G_a, S) \triangle R(G_b, S)$, if $\text{lca}(r(G)) = \text{lca}(b(r(G))) = \text{lca}(a(r(G))) = r(S)$.

Note that Def. 3.3 and Def. 4.1 are trivially equivalent. Next we show that Def. 3.2 and Def. 4.1 are equivalent, thus implying also that Def. 3.2 and Def.3.3 are equivalent.

Theorem 4.13. *Definitions 3.2 and 4.1 are equivalent.*

Proof. Let $R_1(G, S)$ and $R_2(G, S)$ be the reconciled trees constructed by Definitions 3.2 and 4.1 respectively.

Let $k = |L(S)| + |L(G)|$, we will show by induction on k that Definitions 3.2 and 4.1 are equivalent by showing that, given G, S, $R_1(G,S)$ and $R_2(G,S)$ are isomorphic. When $k = 1$, $R_1(G,S) = R_2(G,S) = S$ and the lemma holds.

Assume that the reconciled trees $R_1(G,S)$ and $R_2(G,S)$ for a gene tree G and a species tree S are isomorphic when $|L(G)| + |L(S)| < k$. Next we show that $R_1(G,S)$ and $R_2(G,S)$ are isomorphic when $|L(G)| + |L(S)| = k$.

Observe that condition under which point i, for $i = 1,2,3,4$, of Def. 3.2 applies corresponds to the condition under which point i of Def. 4.1 applies.

Consider point 1 of Def. 3.2 and Def. 4.1 and note that they are equivalent. Moreover since $|L(G)| + |L(S_a)| < k$, by induction $R_1(G,S_a)$ and $R_2(G,S_a)$ are isomorphic, and thus also $R_1(G,S)$, $R_2(G,S)$ are isomorphic.

Note that point 4 of Def. 3.2 and point 4 of Def.4.1 are equivalent. Moreover since $|L(G_a)| + |L(S)| < k$ and $|L(G_b)| + |L(S)| < k$, by induction $R_1(G,S_a)$ and $R_2(G,S_a)$ are isomorphic and, similarly, $R_1(G,S_b)$ and $R_2(G,S_b)$ are isomorphic. Thus also $R_1(G,S)$ and $R_2(G,S)$ are isomorphic.

Consider now point 2 of Def. 3.2 and Def. 4.1. Note that if $s_1 = a(r(S))$ and $s_2 = b(r(S))$, $R_1(G,S) = S(a(r(S)) \to R_1(G_a,S_a), b(r(S)) \to R_1(G_b,S_b))$, which is isomorphic to $R_2(G,S) = R_2(G_a,S_a) \triangle R_2(G_b,S_b)$; in fact, since $|L(G_a)| + |L(S_a)| < k$ and $|L(G_b)| + |L(S_b)| < k$, by induction hypothesis $R_1(G_a,S_a)$, $R_1(G_b,S_b)$ are isomorphic to $R_2(G_a,S_a)$, $R_2(G_b,S_b)$ respectively.

Thus assume $s_1 \subset a(r(S))$ and $s_2 \subset b(r(S))$, observe that $R_1(G,S) = S(s_1 \to R_1(G_a,S_a|s_1), s_2 \to R_1(G_b,S_b|s_2))$. Consider Def. 4.1 and, in particular, consider $R_2(G_a,S_a)$. Note that condition associated with point 1 holds for $R_2(G_a,S_a)$ until it considers $R_2(G_a,S_a|s_1)$, thus obtaining $R_{2a} = S_a(s_1 \to R_2(G_a,S_a))$. Similarly, $R_{2b} = S_b(s_2 \to R_2(G_b,S_b))$. Now since $|L(G_a)| + |L(S|s_1)| < k$, it follows by induction that $R_1(G_a,S|s_1)$ and $R_2(G_a,S|s_1)$ are isomorphic; moreover, since $|L(G_b)| + |L(S|s_2)| < k$, $R_1(G_b,S|s_2)$ and $R_2(G_b,S|s_2)$ are isomorphic. Hence also $R_1(G,S)$ and $R_2(G,S)$ are isomorphic.

Consider now point 3 of Def. 3.2 and point 3 of Def. 4.1. Now if $s_1 = a(r(S))$, $R_1(G,S) = S(a(r(S)) \to R_1(G_a,S)) \triangle R_1(G_b,S)$, which is isomorphic to $R_2(G,S) = R_2(G_a,S) \triangle R_2(G_b,S)$; in fact, since $|L(G_a)| + |L(S)| < k$ and $|L(G_b)| + |L(S)| < k$, by induction hypothesis $R_1(G_a,S)$ and $R_2(G_a,S)$ are isomorphic and, similarly, $R_1(G_b,S)$ and $R_2(G_b,S)$ are isomorphic.

Thus assume $s_1 \subset a(r(S))$ and observe that $R_1(G,S) = S(s_1 \to R_1(G_a,S|s_1)) \triangle R_1(G_b,S|s_2))$. Consider Def. 4.1, and note that for $R_2(G,S)$ the condition associated with point 3 holds until it considers $R_2(G_a,S|s_1)$, thus obtaining $R_{2a} = S(s_1 \to R_2(G_a,S|s_1))$. Now since $|L(G_a)| + |L(S|s_1)| < k$, it follows by induction that $R_1(G_a,S|s_1)$ and $R_2(G_a,S|s_1)$ are isomorphic and thus $R_{1a} = S(s_1 \to R_1(G_a,S|s_1))$ and $R_{2a} = S(s_1 \to R_2(G_a,S|s_1))$ are isomorphic. Moreover, $R_{1b} = R_1(G_b,S_b)$ and $R_{2b} = R_2(G_b,S_b)$ are isomorphic by inductive hypothesis, implying also that $R_1(G,S)$ and $R_2(G,S)$ are isomorphic. □

5 Exemplar-driven reconciliation

In this section, we investigate gene tree reconciliation based on the duplication cost model combined with the notion of exemplar introduced in genome rearrangement [2] to analyze gene families. Exemplar analysis consists of extracting from the various genome sequences a single copy of a gene (called an exemplar) that minimizes a cost function. Following a direction of research proposed and motivated in [2], we introduce the exemplar tree problem, which

naturally derives by investigating a new approach to phylogenetic reconstruction from gene families that combines tree reconstruction and the exemplar analysis. In [7] it is proved that the phylogenetic reconstruction from gene families is an **NP**-hard problem, but the proof requires that the occurrences of labels in G are unbounded. More precisely the problem can be stated as follows. Let $d(G, S)$ be the duplication distance representing the number of duplication induced by lca mapping from the gene tree G to S, the species tree, where $d(G, S) = |\{x \in G | \text{lca}(x) = \text{lca}(c(x)), c(x) \in \{a(x), b(x)\}\}|$.

Problem 1 (Minimum Duplication Problem (MDP)). Given a gene tree G, find a species tree S such that $L(S) = L(G)$ and $d(G, S)$ is minimum.

In this section we investigate the complexity of a variant of MDP obtained by requiring that S must be a homomorphic subtree of G, that is S is obtained from G by extracting a single copy of each label, so that the resulting tree minimizes the number of duplications in S. Given a gene tree G, an *exemplar species tree* for G is any species tree S such that $L(G) = L(S)$ and is a homomorphic subtree of G. The formal definition of the problem follows:

Problem 2 (Exemplar Tree (ET)). Given a gene tree G, find an exemplar species tree S such that the duplication distance $d(G, S)$ is minimized.

Clearly the problem becomes harder as the number of occurrences of a label becomes larger, hence it is interesting to study the computational complexity parametrized by the maximum number of occurrences of a label.

In the following by ET-B, with B integer, we denote the ET problem when the number of copies of a given label in a gene tree is at most B. The main result of this section is proving that ET-2 is **NP**-complete, that is restricting to instances of ET where each label appears at most twice does not help in designing an exact polynomial-time algorithm.

To prove that ET-2 is **NP**-hard we use a reduction from the Vertex Cover problem on cubic graphs [4].

Let $G = (V, E)$ be a cubic graph, with vertex set $V = \{v_1, \cdots, v_n\}$ and edge set $E = \{e_1, \cdots, e_m\}$. Then construct the gene tree T_G over set of labels $L(T_G) = V \cup E \cup L_r \cup L_l \cup \{u\}$, where $L_l = \cup_{1 \leq j \leq n} I_j$, $I_j = \{i_1^j, \cdots, i_{2n}^j\}$, $L_r = \{r_1, \cdots, r_{3n^2}\}$. For each vertex v_i of the graph G, the gene tree T_G contains a subtree T_i. Thus T_G has subtrees $T_1, \cdots, T_n$ as illustrated in Fig. 3(a), where the structure of each tree T_i is shown in Fig. 3(b). Each tree T_i consists of a set of leaves I_i and the subtree T_i' which has leafset $\{e_{i_1}, e_{i_2}, e_{i_3}, v_i\}$, where $e_{i_1}, e_{i_2}, e_{i_3}$ are the edges that are incident on vertex v_i.

We call *left line* and *right line* the paths connecting the root to respectively the leftmost and the rightmost occurrence of the label u of T_G. Both lines are represented as thick lines in Fig. 3(a).

The following properties relate T_G to an exemplar species tree S for T_G. Note that only labels in $E \cup \{u\}$ occur in two copies in T_G. First note that, due to the presence of the two copies of u, the following lemma holds.

Lemma 5.1. *Given a species tree S that is an exemplar species tree for T_G, then either a duplication occurs in all internal nodes of T_G along the* left line *or in all internal nodes of T_G along the* right line.

Proof. Since two copies of label u occur in T_G, either the one on the left line or the one on the right line must be deleted from T_G to obtain an exemplar species tree for T_G. Now, assume

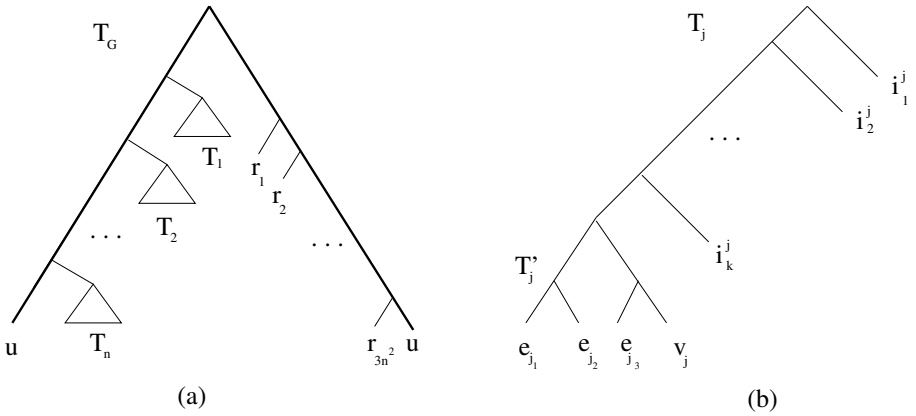

Figure 3: The tree T_G (a), and the tree T_j (b)

that label u is deleted on the left line to obtain a species tree S from T_G. Since for each node x along the left line $x, y \in \mathcal{C}(x)$, where the other copy of label u is on the right subtree of T_G, and y is a node on the left subtree of T_G, being S a homomorphic subtree of T_G, it must be that $\text{lca}(x) = r(T_G)$. □

Consequently, n duplications occur on the left line. Similarly, if label u is deleted on the right line to obtain a subtree S of T_G, then $3n^2$ duplications occur on it, one for each node on the right line.

The following lemma is immediate by the structure of the trees T'_j given in Fig. 3(b) and by definition of duplication.

Lemma 5.2. *Given a species tree S that is an exemplar species tree for T_G, then each tree T'_j of T_G can contain at most one duplication, which is associated with the root r_j of T'_j.*

Lemma 5.3. *Given a species tree S that is an exemplar species tree for T_G, if some label in E of T_j is deleted, then $2n$ or $2n+1$ duplications occur in T_j.*

Proof. Assume that some label in E of the tree T_j is deleted to obtain a subtree S of T_G; then a duplication can be associated with just a single node in T'_j, the root r_j of T'_j. Let x_s be the node of the exemplar species tree S of T_G such that $\text{lca}(r_j) = x_s$. Note that, since S is a subtree of T_G, x_s is a node along the left line of T_G and S. Now, for each internal node y of T_j, with y not in T'_j, $\mathcal{C}(y) = \mathcal{C}(r_j) \cup X$, where $X \subseteq I_j$ and $\mathcal{C}(x_s) \supseteq L(T_j)$, with $L(T_j) \supseteq X$ and $\text{lca}(r_j) = x_s$. Thus, it follows that $\text{lca}(y) = x_s$. But, since there are $2n$ of such internal nodes, in T_j occur at least $2n$ duplications and by Lemma 5.2, at most $2n+1$-duplications. □

Proposition 5.4. *A cubic graph G has a node cover of size C if and only if, given the gene tree T_G, there exists an exemplar species tree S for T_G with $d(T_G, S) \leq C(2n+1) + n$.*

Proof. Notice that an exemplar species tree for T_G must contain a unique copy of each label e_i and of u, which are the only ones that occur twice in T_G. In fact, each edge e of the graph G is incident in two vertices w and v, thus e is a leaf of both trees T_w and T_v.

Now we show that if C vertices are enough to cover all edges in G, then at most $C(2n+1)+n$ duplications occur in T_G. In fact the copy of u that is kept is the one along the right line, moreover

let $N = \{v_{i,1}, \cdots, v_{i,C}\}$ be a vertex cover of G, then each label of an edge in E occurs in some subtree in $\{T_{i,1}, \cdots, T_{i,C}\}$. Thus, at most $C(2n+1)$ duplications are required to delete a copy of each label in E, plus n duplications that occur along the left line.

Vice versa we show that if in an exemplar species tree for T_G at most $C(2n+1)+n$ duplications occur, then G has a node cover of size C. Note that the occurrence of u that must be deleted is the one that is on the left line of T_G, otherwise we would have at least $3n^2$ duplications, that is more than $C(2n+1)+n$. Now, if a given occurrence of a label e_i is deleted in a subtree T'_j of T_j in the gene tree T_G, that is e_i is an edge incident in vertex v_j, by Lemma 5.3, at least $2n$ duplications and at most $2n+1$-duplications occur in T_j.

Indeed, as shown above, since by Lemma 5.3, by deleting one or more label in E in a subtree T_j associated with the vertex v_j at least $2n$ duplications occur, since $2n > C$ it follows that in at most C subtrees T_j of T_G some labels can be deleted, otherwise the number of duplications would be larger than $d(T_G, S) \leq C(2n+1)+n$. By construction this fact implies that C vertices are incident to all edges E of G, thus proving that C is the size of a node cover. □

Theorem 5.5. *The decision version of the ET-2 problem is **NP**-complete.*

Proof. The problem is trivially in **NP**. Furthermore,from Property 5.4 it follows that the decision version of Minimum Vertex Cover on cubic graph is polynomial time reducible to ET-2. Thus the theorem follows. □

6 Mapping-driven reconciliation

In this section we propose a new approach to the reconciliation of a gene tree to a species tree based on the notion of a general mapping as a measure of duplication cost. Biological reasons for adopting a mapping which generalizes the lca have been proposed in [9], though no formal characterization of such mappings has been given.

In this direction our contribution is the definition of two new problems. The first problem generalizes the MDP, where we suggest that the mapping used to count duplications might not be restricted to the lca mapping, but in order to have biological relevance it must preserve cluster inclusion, as defined in the following definition:

Definition 6.1 (Inclusion-preserving mapping). A function δ which maps each node of G to a node of S is called *inclusion-preserving* if for each $x_1, x_2 \in G$, with $L(x_1) \subseteq L(x_2)$, then $\mathcal{C}(x_1) \subseteq \mathcal{C}(\delta(x_1)) \subseteq \mathcal{C}(\delta(x_2))$.

In the following, an inclusion-preserving mapping from G to S is called *mapping* and denoted as δ. The main biological justification for such definition is that any inclusion-preserving mapping associates to node x of a gene tree a set of species that includes all those in the subtree of G rooted at x. Please notice the lca mapping associates with each x the smallest cluster of the species tree with the desired property.

The duplication distance from G to S induced by the mapping δ, denoted by $d_\delta(G, S)$, is the number of duplications induced by δ, where analogously as lca a duplication occurs at node x if and only if $\delta(x) = \delta(a(x))$ or $\delta(x) = \delta(b(x))$.

We are now able to introduce two problems that are quite similar to the Minimum Duplication Problem.

Problem 3 (Minimum Duplication Mapped Tree Problem (MDMT)). Given a gene tree G, compute a species tree S such that $L(S) = L(G)$ and a mapping δ, from G to S, such that $d_\delta(G, S)$ is minimum.

Problem 4 (Minimum Duplication Mapping Problem (MDM)). Given a gene tree G and a species tree S such that $L(G) \subseteq L(S)$, compute a mapping δ from G to S, such that $d_\delta(G, S)$ is minimum.

Clearly the MDMT problem is more general than both the MDM and the MDP problems; more precisely, MDM is the restriction of MDMT where the species tree is given in the instance, while MDP is the restriction of MDMT where the mapping is restricted to be the lca mapping. Moreover please notice that any solution of the MDP problem is a feasible, but not necessarily optimal, solution of the MDM problem.

In the following we give an efficient algorithm for solving MDM via dynamic programming. The following property is central to our algorithm.

Proposition 6.1. *Let G and S be respectively a gene tree and a species tree, then there exists an optimal solution δ of MDM such that for each node g of G either $\delta(G_a) = \delta(G_b) = \delta(g)$ or $\delta(G_a) \neq \delta(g) \neq \delta(G_b)$.*

Proof. Let δ be an optimal solution of MDM. For each node g such that exactly one of its children is mapped in $\delta(g)$, we modify the mapping both children of g by posing $\delta(G_a) = \delta(G_b) = \delta(g)$. It is immediate to note that the number of duplications is not increased, consequently the new solution is still optimal. □

In order to describe a dynamic programming algorithm for the MDM problem we need to point out a recursive definition of an optimal solution. In fact, given an optimal solution $opt(G, S)$, there are two possible cases: either both children of $r(G)$ are mapped in $\delta(r(g))$, or no children of $r(G)$ are mapped in $\delta(r(g))$. Let y be the node where $r(G)$ is mapped. In the first case an optimal solution is obtained from optimal solutions $opt(a(G), S)$ and $opt(b(G), S)$, with cost $cost(opt(a(G), S))+cost(opt(b(G), S))+1$, while in the second case an optimal solution is obtained from optimal solutions $opt(a(G), a(S))$ and $opt(b(G), b(S))$ with cost $cost(opt(a(G), a(S)))+cost(opt(b(G), b(S)))$.

We have then the following recurrence for $cost(opt(G, S))$:

$$\begin{cases} cost(opt(G|x_1, S)) + cost(opt(G|x_2, S)) + 1 & \text{if} \quad \delta(x_1) = \delta(x_2) = r(S) \\ cost(opt(G|x_1, S|y_1)) + cost(opt(G|x_2, S|y_2)) & \text{if} \quad \delta(x_1) \neq r(S) \text{ and } \delta(x_2) \neq r(S) \end{cases}$$

To prove the correctness of the recurrence we consider the two possible cases.

Case 1. Assume that an optimal solution δ maps $a(r(G))$ and $b(r(G))$ in $r(S)$, i.e. there is a duplication in $r(G)$.

Let δ_1 be a solution obtained from the recurrence above. A duplication occurs in the root $r(G)$, and the other duplications for G will occur in the left subtree $a(G)$ and in the right subtree $b(G)$. Clearly if $cost(\delta) \geq cost(\delta_1)$ the proof is completed, hence assume now that $cost(\delta) < cost(\delta_1)$. Since in both δ and δ_1 a duplication is placed in the root, the solution δ places in G_a or in G_b strictly less duplications than the number of duplications placed by δ_1 in the same subtrees, hence the optimality of G_a or G_b is violated, which is a contradiction.

Case 2. Assume that an optimal solution δ maps $a(r(G))$ and $b(r(G))$ in two nodes different from $r(S)$, i.e. there is no duplication in $r(G)$.

Let δ_1 be a solution obtained from the recurrence above. The duplications from G to S will be placed in the left subtree $a(G)$ and in the right subtree $b(G)$, since there are no duplications in the root.

Clearly if $cost(\delta) \geq cost(\delta_1)$ the proof is completed, hence assume now that $cost(\delta) < cost(\delta_1)$. Along the same lines as for case 1, w.l.o.g. we can assume that the cost of δ in $a(G)$ is less than the cost of δ_1 in $a(G)$. But then the optimality of $a(G)$ would be violated.

The algorithm mainly consists of filling in a bidimensional table indexed by the nodes of G and the nodes of S. In the following we will denote each cell of such table by $T[g, s]$ representing the cost of an optimal solution over the instance $(G|g, S|s)$. We order the nodes of each tree G and S with a postorder visit, that is according to the ancestral relation between nodes where each node x comes after all nodes (different from x) that are in the subtree rooted at x. Then the table $T[g, s]$ is filled respecting the ordering of nodes, that is starting from the leaves and going towards the root s. Clearly if S consists of only one leaf the only possible solution is mapping each node g to the leaf of S, consequently, $T[g, s] = 0$ if both g and s are leaves with the same label, while $T[g, s] = \infty$ if g and s are leaves with different labels. The recurrence for $T[g, s]$ when g and s are not both leaves is described as follows, where all possible mappings from the two topmost levels of G and S are stated:

$$T[g,s] = \min \begin{cases} T[a(g),s] + T[b(g),s] + 1 & \text{if } \mathcal{C}(G|g) \subseteq \mathcal{C}(S|s) \\ T[a(g),a(s)] + T[b(g),b(s)],\ T[a(g),b(s)] + T[b(g),a(s)] & \text{if } \mathcal{C}(G|g) \subseteq \mathcal{C}(S|s) \\ T[a(g),a(s)] + T[b(g),a(s)] & \text{if } \mathcal{C}(G|g) \subseteq \mathcal{C}(S|a(s)) \\ T[a(g),b(s)] + T[b(g),b(s)] & \text{if } \mathcal{C}(G|g) \subseteq \mathcal{C}(S|b(s)) \\ T[g,a(s)],\ T[g,b(s)] & \end{cases}$$

The above recurrence, whose correctness follows from Prop. 6.1, can be implemented easily by filling in the entries according to the ordering of nodes such that at the end the entry $T[r(G), r(S)]$ contains the optimum number of duplications. Providing appropriate backpointer information has been saved during the fill-in procedure, traceback from this entry can then be used to compute the optimal solution.

7 Open problems

The MDMT problem is more general than MDP, but the **NP**-hardness of MDP does not immediately extend to MDMT, since the sets of feasible solutions of the two problems are not comparable. Consequently an interesting open question is whether there exists an efficient algorithm for the MDMT problem. As far as we know, it is still a challenging open question whether MDT is polynomial time solvable or not in the case that G is uniquely labeled or the number of occurrences of each label is bounded and our Theorem 5.5 is only a preliminary step in settling the computational complexity of MDT.

8 Acknowledgements

This work has been supported by FIRB 2003 grant *Bioinformatics: genomics and proteomics*.

References

[1] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 2nd edition, 2001.

[2] N. El-Mabrouk and D. Sankoff. Duplication, rearrangement and reconciliation. *Comparative Genomics: Empirical and Analytical Approaches to Gene Order Dynamics, Map alignment and the Evolution of Gene Families. Computational Biology Series*, 1:537–550, 2000.

[3] M. Fellows, M. Hallett, and U. Stege. On the multiple gene duplication problem. In *Proceedings of the 9th International Symposioum on Algorithms and Computation (ISAAC98)*, 1998.

[4] M. Garey and D. Johnson. *Computer and Intractability: A Guide to the Theory of NP-completeness*. W. H. Freeman, 1979.

[5] M. Goodman, J. Czelusniak, G. Moore, A. Romero-Herrera, and G. Matsuda. Fitting the gene lineage into its species lineage, a parsimony strategy illustrated by cladograms constructed from globin sequences. *Systematic Zoology*, 28:132–163, 1979.

[6] R. Guigò, I. Muchnik, and T. Smith. Reconstruction of ancient molecular phylogeny. *Mol. Phy. and Evol.*, 6(2):189–213, 1996.

[7] B. Ma, M. Li, and L. Zhang. From gene trees to species trees. *SIAM Journal on Computing*, 30(3):729–752, 2000.

[8] R. Page. Maps between trees and cladistic analysis of historical associations among genes. *Systematic Biology*, 43:58–77, 1994.

[9] R. M. Page and J. Cotton. Vertebrate phylogenomics: reconciled trees and gene duplications. In *Proceedings of Pacific Symposium on Biocomputing 2002 (PSB2002)*, pages 536–547, 2002.