Journal of Bioinformatics and Computational Biology

Reconstructing Protein and Gene Phylogenies using reconciliation and soft-clustering

Esaie Kuitche

Department of Computer Science, Université de Sherbrooke
Sherbrooke, QC J1K2R1, Canada
esaie.kuitche.kamela@USherbrooke.ca

Manuel Lafond

Department of Mathematics and Statistics, University of Ottawa
Ottawa, ON K1N6N5, Canada
mlafond2@UOttawa.ca

Aïda Ouangraoua

Department of Computer Science, Université de Sherbrooke
Sherbrooke, QC J1K2R1, Canada
aida.ouangraoua@USherbrooke.ca

Received (Day Month Year)
Revised (Day Month Year)
Accepted (Day Month Year)

The architecture of eukaryotic coding genes allows the production of several different protein isoforms by genes. Current gene phylogeny reconstruction methods make use of a single protein product per gene, ignoring information on alternative protein isoforms. These methods often lead to inaccurate gene tree reconstructions that require to be corrected before phylogenetic analyses. Here, we propose a new approach for the reconstruction of gene trees and protein trees accounting for alternative protein isoforms. We extend the concept of reconciliation to protein trees, and we define a new reconciliation problem called MinDRGT that consists in finding a gene tree that minimizes a double reconciliation cost with a given protein tree and a given species tree. We define a second problem called MinDRPGT that consists in finding a protein supertree and a gene tree minimizing a double reconciliation cost, given a species tree and a set of protein subtrees. We propose a shift from the traditional view of protein ortholog groups as hard-clusters to soft-clusters and we study the MinDRPGT problem under this assumption. We provide algorithmic exact and heuristic solutions for versions of the problems, and we present the results of applications on protein and gene trees from the Ensembl database. The implementations of the methods are available at https://github.com/UdeS-CoBIUS/Protein2GeneTree and https://github.com/UdeS-CoBIUS/SuperProteinTree.

Keywords: Protein Tree; Gene Tree ; Orthology; Reconciliation; Soft-clustering

1. Introduction

Recent genome analyses have revealed the ability of eukaryotic coding genes to produce several transcripts and proteins isoforms. This mechanism plays a major role in the functional diversification of genes [19,23]. Still, current gene phylogeny reconstruction methods make use of a single protein product per gene that is usually the longest protein called the "reference protein" ignoring the production of alternative protein isoforms [1,27,30]. It has been shown that these sequence-based methods often return incorrect gene trees [15,27]. Thus, several methods have been proposed for the correction of gene trees [24,31]. Recently, a few models and algorithms aimed at reconstructing the evolution of full sets of gene products along gene trees were introduced [7,32]. Some models have also been proposed to study the evolution of alternative splicing and gene exon-intron structures along gene trees [17,19]. All these models require the input of accurate gene trees and are biased when the input gene trees contain errors. Here, we explore a new approach in order to directly reconstruct accurate gene phylogenies and protein phylogenies while accounting for the production of alternative protein isoforms by genes. We introduce new models and algorithms for the reconstruction of gene phylogenies and full sets of proteins phylogenies using *reconciliation* [10] and *soft-clustering*. Reconciliation, first introduced by Goodman et al. in 1979 [11], is a widely used tool to explain the incongruence between a gene tree and a species tree (see e.g. [2,4,5,6,12,13,21,22] and [9] for a survey on reconciliation algorithms). Here, we also use reconciliation to compare protein trees and gene trees.

First, we present a model of protein evolution along a gene tree that involves two types of evolutionary events called *protein creation* and *protein loss*, in addition to the classical evolutionary events of speciation, gene duplication and gene loss considered in gene-species tree reconciliation. Second, we propose an extension of the framework of gene-species tree reconciliation in order to define the concept of protein-gene tree reconciliation, and we introduce new reconciliation problems aimed at reconstructing optimal gene trees and proteins trees. We define the problem of finding a gene tree minimizing the sum of the protein-gene and gene-species reconciliation costs, given the protein tree and the species tree. We call this problem the *Minimum Double Reconciliation Gene Tree* (MinDRGT) problem. We also define the problem of jointly finding a protein supertree and a gene tree minimizing the sum of the protein-gene and gene-species reconciliation costs, given the species tree and a set of subtrees of the protein tree to be found. We call this problem the *Minimum Double Reconciliation Protein and Gene Tree* (MinDRPGT) problem. Third, we define an adaptation of the UPGMA algorithm for the soft-clustering of proteins into ortholog groups and we use it for the reconstruction of protein supertrees.

The paper is organized as follows. We first formally define, in Section 3, the new protein evolutionary models and the related reconciliation problems, MinDRGT and MinDRPGT, for the reconstruction of gene phylogenies and full sets

of protein phylogenies. In Section 4, we prove the NP-hardness of some versions of MinDRGT, especially the one called MinDRGT$_{CD}$ that consists in minimizing the number of protein creation and gene duplication events. Next, in Section 5, we consider the MinDRGT problem in a special case where each gene is associated to a single protein. This restriction is relevant for the correction of gene trees output by sequence-based gene phylogeny reconstruction methods using a single protein per gene. Such methods make the unsupported assumption that each pair of leaf proteins in the protein tree is related through a least common ancestral node that corresponds to a speciation or a gene duplication event, and then, they output a gene tree equivalent to the reconstructed protein tree. In this perspective, the MinDRGT problem under the restriction that each gene is associated to a single protein, allows pairs of proteins to be related through ancestral protein creation events, and then asks to find an optimal gene tree, possibly different from the input protein tree. In other terms, the protein tree is not confused with the gene tree, but it is used, together with the species tree, to guide the reconstruction of the gene tree. We first show that, even with the restriction that each gene is associated to a single protein, for most versions of the MinDRGT problem, the optimal gene tree may differ from the input protein tree. In particular, we give a counterexample that shows that the duplication cost, the lost cost and the mutation cost considered in the classical gene-species tree reconciliation do not satisfy the triangle inequality. We then exhibit a heuristic algorithm called Protein2GeneTree for the MinDRGT problem that consists in building the optimal gene tree by applying modifications on the input protein tree guided by the species tree.

In Section 6, we consider the MinDRPGT problem aimed at jointly reconstructing both a protein phylogeny and a gene phylogeny. We consider a restriction on the input data that requires the set of input protein subtrees to be the set of all inclusion-wise maximum subtrees of the target protein supertree P that contain no protein creation node. Such an input consists of phylogenetic trees on clusters of orthologous proteins that can be obtained by using a soft-clustering approach to group proteins. Under this assumption, we present a polynomial-time exact algorithm called SuperProteinTree for computing the target protein supertree P, which allows to reduce MinDRPGT to a special case of MinDRGT where the input protein tree P is given with a partial labeling of its nodes. The algorithm consists in first reconstructing a partition of P composed of subtrees that are either entirely included or excluded in each input subtree, and then combining these partition subtrees into P.

In Section 7, the results of applying Protein2GeneTree for the correction of gene trees from the Ensembl database [16] show that the new framework allows to reconstruct gene trees whose double reconciliation costs are decreased, as compared to the initial Ensembl gene trees [30]. We also show that most (>80%) corrected trees cannot be rejected on a statistical basis as compared to the initial trees, according to the Approximately Unbiased (AU) Test [28]. Next, we present our modified UPGMA algorithm for protein soft-clustering and the results of applying it together with the

algorithm SuperProteinTree for the construction of protein supertrees. The results show that the reconstructed protein supertree supports the hypothesis of protein creation events that happened in ancestral genes with groups of orthologous protein isoforms shared by several extant genes.

2. Preliminaries: protein trees, gene trees and species trees

In this section, we introduce some preliminary notations: $\mathcal{S}$ denotes a set of species, $\mathcal{G}$ a set of genes representing a gene family, and $\mathcal{P}$ a set of proteins produced by the genes of the gene family. The three sets are accompanied with a mapping function $s : \mathcal{G} \to \mathcal{S}$ mapping each gene to its corresponding species, and a mapping function $g : \mathcal{P} \to \mathcal{G}$ mapping each protein to its corresponding gene. In the sequel, we assume that $\mathcal{S}$, $\mathcal{G}$ and $\mathcal{P}$ satisfy $\{s(x) : x \in \mathcal{G}\} = \mathcal{S}$ and $\{g(x) : x \in \mathcal{P}\} = \mathcal{G}$, without explicitly mentioning it.

Phylogenetic trees: A tree T for a set L is a rooted binary tree whose leafset is L. The leafset of a tree T is denoted by $\mathcal{L}(T)$ and the set of nodes of T is denoted by $\mathcal{V}(T)$. Given a node x of T, the complete subtree of T rooted at x is denoted by $T[x]$. The *lowest common ancestor* (lca) in T of a subset L' of $\mathcal{L}(T)$, denoted by $lca_T(L')$, is the ancestor common to all nodes in L' that is the most distant from the root of T. $T|_{L'}$ denotes the tree for L' obtained from $T[lca_T(L')]$ by removing every node that does not have a descendant in L', then contracting the nodes with a single child until none remains. Given an internal node x of T, the children of x are arbitrarily denoted by x_l and x_r.

Proteins, genes, and species trees: In the sequel, S denotes a species tree for the set $\mathcal{S}$, G denotes a gene tree for the set $\mathcal{G}$, and P denotes a protein tree for the set $\mathcal{P}$. The mapping function s is extended to be defined from $\mathcal{V}(G)$ to $\mathcal{V}(S)$ such that if x is an internal node of G, then $s(x) = lca_S(\{s(x') : x' \in \mathcal{L}(G[x])\})$, i.e. the image of a node $x \in \mathcal{V}(G)$ in $\mathcal{V}(S)$ is the lca in the tree S of all the images of the leaves of $G[x]$ by s. Similarly, the mapping function g is extended to be defined from $\mathcal{V}(P)$ to $\mathcal{V}(G)$ such that if x is an internal node of P, then $g(x) = lca_G(\{g(x') : x' \in \mathcal{L}(P[x])\})$.

Gene-species tree reconciliation: Each internal node of the species tree S represents an ancestral species at the moment of a speciation event (*Spec*) in the evolutionary history of $\mathcal{S}$. The gene tree G represents the evolutionary history of the genes of the gene family $\mathcal{G}$, and each internal node of G represents an ancestral gene at the moment of a *Spec* or a gene duplication event (*Dup*).

The *LCA-reconciliation* of G with S is a labeling function l_G from $\mathcal{V}(G)-\mathcal{L}(G)$ to $\{Spec, Dup\}$ such that the label of an internal node x of G is $l_G(x) = Spec$ if $s(x) \neq s(x_l)$ and $s(x) \neq s(x_r)$, and $l_G(x) = Dup$ otherwise (see e.g. [4,5,6,10,12,13,21,22]). The LCA-reconciliation induces gene loss events on edges of G as follows: given an edge (x, y) of the tree G such that $y = x_l$ or $y = x_r$, a gene loss event is induced on (x, y)

for each node located on the path between $s(x)$ and $s(y)$ in S (excluding $s(x)$ and $s(y)$). If $l_G(x) = Dup$ and $s(x) \neq s(y)$, an additional loss event preceding all other loss events is induced on (x, y) for $s(x)$. Figure 1 presents a gene tree G on a gene family $\mathcal{G} = \{a_2, a_3, b_0, b_1, b_2, b_3, c_1, c_2, c_3, d_3\}$ reconciled with a species tree S on a set of species $\mathcal{S} = \{a, b, c, d\}$.

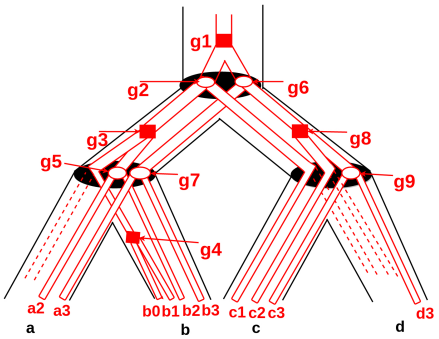


Fig. 1: A gene tree G on a gene family $\mathcal{G} = \{a_2, a_3, b_0, b_1, b_2, b_3, c_1, c_2, c_3, d_3\}$ such that $s(x_i) = x$ for any gene $x_i \in \mathcal{G}$ and species $x \in \mathcal{S} = \{a, b, c, d\}$. The species tree S is $((a, b), (c, d))$. G is reconciled with S: a speciation node x of G is located inside the node $l_G(x)$ of S, and a duplication node x is located on the edge $(p, l_G(x))$ of S such that p is the parent of $l_G(x)$ in S. The gene tree G contains 9 ancestral nodes: g_1, g_3, g_4, g_8 are duplications represented as squared nodes and g_2, g_5, g_6, g_7, g_9 are speciations represented as circular nodes. G contains 3 loss events whose locations are indicated with dashed edges. The same labeled gene tree G is represented in Figure 3 (Top), not embedded in S.

The LCA-reconciliation l_G suggests three possible costs of reconciliation $C_{G \to S}$ between G and S. The *duplication cost* denoted by $D(G, S)$ is the number of nodes x of G such that $l_G(x) = Dup$. The *loss cost* denoted by $L(G, S)$ is the overall number of loss events induced by l_G on edges of G. The *mutation cost* denoted by $M(G, S)$ is the sum of the duplication cost and the loss cost induced by l_G. In the example depicted in Figure 1, the duplication cost is 4, while the loss cost is 3, and the mutation cost is 7.

Homology relations between genes: Two genes x and y of the set $\mathcal{G}$ are called *orthologs* if $l_G(lca_G(\{x, y\})) = Spec$, and *paralogs* otherwise.

3. Model of protein evolution along a gene tree and problem statements

In this section, we first formally describe the new model of protein evolution along a gene tree. Next, we describe an extension of the framework of phylogenetic tree

reconciliation that makes use of the new model, and we state new optimization problems related to the extended framework.

Protein evolutionary model: The protein evolutionary model that we propose is based on the idea that the set of all proteins $\mathcal{P}$ produced by a gene family $\mathcal{G}$ have derived from a set $\mathcal{A}_P$ of common ancestral proteins that were produced by the ancestral gene located at the root of the gene tree G. This ancestral set of proteins evolved along the gene tree through different types of evolutionary and modification events including the classical events of speciation, gene duplication and gene loss. In the sequel, we consider that the ancestral set of proteins $\mathcal{A}_P$ is composed of a single ancestral protein that is the root of a tree for the set of proteins $\mathcal{P}$, but all definitions can be directly extended to protein forests, i.e sets of independent proteins trees rooted at multiple ancestral proteins.

A *protein tree* P is a tree for the set of proteins $\mathcal{P}$ representing the phylogeny of the proteins in $\mathcal{P}$. Each internal node of P represents an ancestral protein at the moment of a Spec, Dup, or a *protein creation event* (*Creat*). A protein creation event represents the appearance of a new protein isoform at a moment of the evolution of a gene family on an edge of the gene tree G. This evolutionary model is supported by recent studies on the evolution of gene alternative splicing patterns and inter-species comparison of gene exon-intron structures [17,19]. In particular, these studies have highlighted that alternative splicing patterns may be gene-specific or shared by groups of homologous genes [3,25]. A protein creation event thus leads to the observation of conserved protein isoforms called *orthologous splicing isoforms* [32] in a group of homologous extant genes descending from the ancestral gene that underwent the protein creation event. Based on these observations, the present model of protein evolution allows to describe the evolution of the full set of proteins produced by a gene family along the gene tree of the family. Figure 2 presents an example of labeled protein tree for a set of proteins $\mathcal{P} = \{a21, a31, b01, b02, b11, b21, b31, c11, c12, c21, c31, d31\}$.

Protein-gene tree reconciliation: We naturally extend the concept of reconciliation to protein trees as follows. The *LCA-reconciliation* of P with G is a labeling function l_P from $\mathcal{V}(P) - \mathcal{L}(P)$ to $\{Spec, Dup, Creat\}$ that labels an internal node x of P as $l_P(x) = Spec$ if $g(x) \neq g(x_l)$ and $g(x) \neq g(x_r)$ and $l_G(g(x)) = Spec$, else $l_P(x) = Dup$ if $g(x) \neq g(x_l)$ and $g(x) \neq g(x_r)$ and $l_G(g(x)) = Dup$, and $l_G(x) = Creat$ otherwise. Note that, if x is such that $\{g(y)|y \in \mathcal{L}(P[x_l])\} \cap \{g(y)|y \in \mathcal{L}(P[x_r])\} \neq \emptyset$, then $l_P(x) = Creat$, and x is called an *apparent creation* node.

Similarly to the LCA-reconciliation l_G, the LCA-reconciliation l_P induces protein loss events on edges of P as follows: given an edge (x, y) of P such that $y = x_l$ or $y = x_r$, a protein loss event is induced on (x, y) for each node located on the path between $g(x)$ and $g(y)$ in G. If $l_P(x) = Creat$ and $g(x) \neq g(y)$, an additional protein loss event preceding all other protein loss events is induced on (x, y) for $g(x)$. A protein loss event corresponds to the loss of the ability to produce a protein

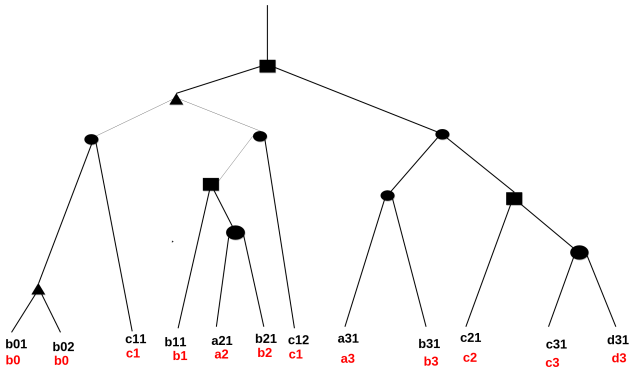


Fig. 2: A protein tree P on the set $\mathcal{P} = \{a21, a31, b01, b02, b11, b21, b31, c11, c12, c21, c31, d31\}$. The nodes of the tree are labeled as speciation (circular nodes), gene duplication (squared nodes), of protein creation events (triangular nodes). For each protein leaf x_{ij} of P, the corresponding gene $x_i = g(x_{ij})$ is indicated below the protein. The LCA-reconciliation that resulted in the labeling of the nodes of P is illustrated in Figure 3.

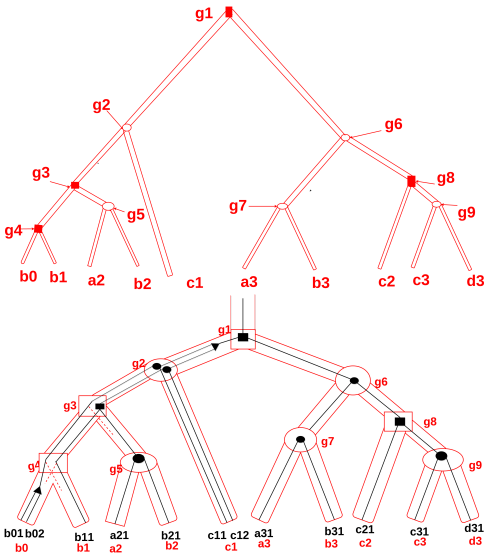


Fig. 3: Top. The labeled gene tree G of Figure 1. Bottom. The protein tree P of Figure 2 reconciled with G. For each internal node x of P, the corresponding image $g(x)$ in G is indicated. The protein tree P contains 2 protein creation nodes (triangular nodes), 3 gene duplication nodes (squared nodes), 6 speciation nodes (circular nodes), and 3 protein loss events indicated as dashed lines.

isoform for an ancestral gene at a moment of the evolution of the gene family.

We define the following three costs of reconciliation $C_{P\to G}$ induced by the LCA-reconciliation l_P of P with G. The *creation cost* denoted by $C(P,G)$ is the number of nodes x of P such that $l_P(x) = Creat$. The *loss cost* denoted by $L(P,G)$ is the overall number of loss events induced by l_P on edges of P. The *mutation cost* denoted by $M(P,G)$ is the sum of the creation cost and the loss cost induced by l_P. In the example depicted in Figure 3, the creation cost is 2, while the loss cost is 3, and the mutation cost is 5.

Homology relations between proteins: Based on the LCA-reconciliation of P with G, we can define the following homology relations between proteins of the set $\mathcal{P}$. Two proteins x and y of $\mathcal{P}$ are called *orthologs* if $l_P(lca_P(\{x,y\})) \neq Creat$, and in this case, we distinguish two types of orthology relationship: x and y are *ortho-orthologs* if $l_P(lca_P(\{x,y\})) = Spec$, and *para-orthologs* otherwise. Note that if x and y are ortho-orthologs (resp. para-orthologs), the genes $g(x)$ and $g(y)$ are orthologs (resp. paralogs). Finally, x and y are *paralogs* if $l_P(lca_P(\{x,y\})) = Creat$. Given a subset L' of $\mathcal{L}(P)$ such that any pair of proteins $(x,y) \in L' \times L'$ are orthologs, the tree $P|_{L'}$ induced by L' is called a *creation-free subtree* of P.

Problem statements: Given a protein tree P, a gene tree G and a species tree S, the *double reconciliation cost* of G with P and S is the sum of a cost $C_{P\to G}$ of reconciliation of P with G and a cost $C_{G\to S}$ of reconciliation of G with S.

Depending on the costs of reconciliation $C_{P\to G}$ considered for P with G, and $C_{G\to S}$ considered for G with S, nine types of double reconciliation cost can be defined. They are denoted by $XY(P,G,S)$ where X is either C for $C(P,G)$ or L for $L(P,G)$ or M for $M(P,G)$, and Y is either D for $D(G,S)$ or L for $L(G,S)$ or M for $M(G,S)$. For example, $CD(P,G,S)$ considers the creation cost for $C_{P\to G}$ and the duplication cost for $C_{G\to S}$.

The definition of the double reconciliation cost naturally leads to the definition of our first reconciliation problem that consists in finding an optimal gene tree G, given a protein tree P and a species tree S.

Minimum Double Reconciliation Gene Tree Problem (MinDRGT$_{XY}$):
Input: A species tree S for $\mathcal{S}$; a protein tree P for $\mathcal{P}$; a gene family $\mathcal{G}$.
Output: A gene tree G for $\mathcal{G}$ that minimizes the double reconciliation cost $XY(P,G,S)$.

The problem MinDRGT assumes that the protein tree P is known, but in practice, phylogenetic trees on full sets of proteins are not always available. Furthermore, the application of sequence-based phylogenetic reconstruction methods for constructing protein trees with more than one protein for some genes is likely to lead to incorrect trees, as for the reconstruction of single-protein-per-gene trees [15,27]. However, proteins subtrees of P can be obtained by building phylogenetic

trees for sets of orthologous protein isoforms [32]. Such subtrees can then be combined in order to obtain the full protein tree P. One way to combine the orthologous protein subtrees consists in following an approach, successfully used in [20] for combining a set of gene subtrees into a gene tree. It consists in jointly reconstructing the combined protein tree P and the gene tree G while seeking to minimize the double reconciliation cost of G with P and S. We then define a second problem that consists in finding an optimal pair of protein tree P and gene tree G, given a species tree S and a set of known subtrees $P_{i,1\leq i\leq k}$ of P.

Minimum Double Reconciliation Protein and Gene Tree Problem (MinDRPGT$_{XY}$):

Input: A species tree S for $\mathcal{S}$; a set of proteins $\mathcal{P}$, a set of subsets $\mathcal{P}_{i,1\leq i\leq k}$ of $\mathcal{P}$ such that $\bigcup_{i=1}^{k}\mathcal{P}_i = \mathcal{P}$, and a set of protein trees $P_{i,1\leq i\leq k}$ such that for each $i, 1 \leq i \leq k$, P_i is a tree for $\mathcal{P}_i$ and a subtree of the target (real) protein tree.

Output: A protein tree P for $\mathcal{P}$ such that $\forall i$, $P|_{\mathcal{P}_i} = P_i$ and a gene tree G for $\mathcal{G} = \{g(x) : x \in \mathcal{P}\}$ that minimize the double reconciliation cost $XY(P, G, S)$.

4. NP-hardness of MinDRGT

In this section, we prove the NP-hardness of MinDRGT$_{XY}$ for $X = C$ and $Y \in \{D, L, M\}$.

Proposition 1. *Given a protein tree P on $\mathcal{P}$ and a gene tree G on $\mathcal{G}$ with a protein-species mapping g, let G' be a gene tree on $\mathcal{G}' = \mathcal{P}$, and S' a species tree on $\mathcal{S}' = \mathcal{G}$ with a gene-species mapping $s = g$. The reconciliation costs from P to G, and from G' to S' satisfy the following: (1) $C(P, G) = D(G', S')$; (2) $L(P, G) = L(G', S')$; (3) $M(P, G) = M(G', S')$.*

From Proposition 1, all algorithmic results for the reconciliation problems between gene and species trees can be directly transferred to the equivalent reconciliation problems between protein and gene trees. In particular, in [22], it is shown that, given a gene tree G, finding a species tree S minimizing $D(G, S)$ is NP-hard. To our knowledge the complexity for the same problem with the $L(G, S)$ or $M(G, S)$ costs are still open, though we believe it is also NP-hard since they do not seem easier to handle than the duplication cost. Theorem 1 uses these results to imply the NP-hardness of some versions of MinDRGT.

Theorem 1. *Suppose that the problem of finding a species tree S minimizing the cost $Y'(G, S)$ with a given gene tree G is NP-hard for $Y' \in \{D, L, M\}$. Let $X = C$ if $Y' = D$, and $X = Y'$ if $Y' \in \{L, M\}$.*

Then for any reconciliation cost function $Y \in \{D, L, M\}$ and a given protein tree P and species tree S, the problem of finding a gene tree G minimizing the double-reconciliation cost $XY(P, G, S)$ is NP-hard, even if $|\mathcal{G}| = |\mathcal{S}|$.

Proof. Let $\{\hat{\mathcal{G}}, \hat{G}, \hat{\mathcal{S}}, \hat{s}\}$ be an instance of the problem of finding an optimal species

tree minimizing Y', such that $\hat{\mathcal{G}}$ is the set of genes, $\hat{G}$ the gene tree, $\hat{\mathcal{S}}$ the set of species, and $\hat{s}$ is the gene-species mapping. We will reduce this problem to that of minimizing the double reconciliation cost for an instance $\{\mathcal{P}, P, \mathcal{G}, \mathcal{S}, S, g, s\}$. We create the protein set $\mathcal{P}$, protein tree P, species set $\mathcal{S}$ and species tree S with protein-gene mapping g and gene-species mapping s, as follows: $\mathcal{G} = \hat{\mathcal{S}}$ (i.e. the species become the genes), and $\mathcal{S}$ is such that $|\mathcal{S}| = |\mathcal{G}|$, and the mapping s a bijection between $\mathcal{S}$ and $\mathcal{G}$ (i.e. each species has one gene). The tree S is an arbitrary tree over leafset $\mathcal{S}$. Denote $n = |\mathcal{S}| = |\mathcal{G}|$. The protein set $\mathcal{P}$ consists in n^3 copies of $\hat{\mathcal{G}}$, i.e. $\mathcal{P} = \bigcup_{\hat{g}\in\hat{\mathcal{G}}}\{g_1, g_2, \ldots, g_{n^3}\}$. For each protein $g_i \in \mathcal{P}$ corresponding to gene $\hat{g} \in \hat{\mathcal{G}}$, we set the mapping $g(g_i) = \hat{s}(\hat{g})$. To construct the protein tree P, first create a set of n^3 copies $\mathbb{P} = \{P_1, \ldots, P_{n^3}\}$ of $\hat{G}$ such that for each $i \in [n^3]$, the tree P_i is obtained from $\hat{G}$ by replacing each gene $\hat{g} \in \mathcal{L}(\hat{G})$ by its i-th corresponding protein $g_i \in \mathcal{P}$. Next, let T be any binary tree with n^3 leaves $l_1, \ldots, l_{n^3}$, and for each $i \in [n^3]$, replace l_i by the tree P_i. Note that every of the $n^3 - 1$ internal node x initially in T must be an apparent creation node. Also note that no matter what the gene tree G is, x will be mapped to the root of G, as well as its two children x_l and x_r, implying that there are no losses on a branch incident to a node initially in T. The hardness of MinDRGT$_{XY}$ follows from the next (crude) upper bound:

Claim 1. *Let T_1, T_2 be two trees on the same leafset L. Then for any cost $Z \in \{D, L, M\}$, $Z(T_1, T_2) \leq 5|L|^2$.*

Proof. It suffices to prove the claim for the mutation cost M. When reconciled with T_2, the tree T_1 has at most $|\mathcal{V}(T_1)\setminus\mathcal{L}(T_1)| \leq |L|$ duplication nodes. As for the losses, each branch of T_1 induces at most $|\mathcal{V}(T_2)| \leq 2L$ losses, and so T_1 has a total of at most $|E(T_1)|2L \leq 2|L|2|L| = 4|L|^2$ losses. Thus $M(T_1, T_2) \leq 4|L|^2 + |L| \leq 5|L|^2$. □

We now show that there is a species tree $\hat{S}$ with $Y'(\hat{G}, \hat{S}) \leq k$ if and only if there is a gene tree G with double reconciliation cost $XY(P, G, S) \leq k(n^3 + 1) + 5n^2$.

($\Rightarrow$): let $\hat{S}$ be such that $Y'(\hat{G}, \hat{S}) \leq k$. Then our solution is $G = \hat{S}$. Since there are no losses on the branches incident to the nodes initially in T, and because each subtree P_i of $\mathbb{P}$ is a copy of $\hat{G}$, we have $X(P, G) = \sum_{P_i\in\mathbb{P}} X(P_i, G) + n^3 - 1 = \sum_{P_i\in\mathbb{P}} Y'(\hat{G}, \hat{S}) + n^3 - 1 \leq n^3(k + 1)$. Moreover, by Claim 1, $Y(G, S) \leq 5n^2$, and so the double reconciliation cost is at most $n^3(k + 1) + 5n^2$, as desired.

($\Leftarrow$): let G be a gene tree with $XY(P, G, S) \leq (k + 1)n^3 + 5n^2$. We claim that letting $\hat{S} = G$, we obtain a solution with $Y'(\hat{G}, \hat{S}) \leq k$. Suppose otherwise that $Y'(\hat{G}, \hat{S}) \geq k + 1$. Then as each P_i is a copy of $\hat{G}$ and $\hat{S} = G$, $\sum_{P_i\in\mathbb{P}} X(P_i, G) = \sum_{P_i\in\mathbb{P}} Y'(\hat{G}, \hat{S}) \geq (k + 1)n^3$. Moreover, there are $n^3 - 1$ creation nodes above the P_i's, and so $X(P, G) \geq (k + 2)n^3 - 1$. But for large enough n (i.e. $n \geq 6$), $(k+2)n^3 - 1 > (k+1)n^3 + 5n^2$, contradicting our assumption on $XY(P, G, S)$. This completes the proof.

Corollary 1. *The* MinDRGT$_{XY}$ *problem is NP-hard for $X = C$ and $Y \in \{D, L, M\}$.*

5. MinDRGT for the case $\mathcal{P} \Leftrightarrow \mathcal{G}$

In Section 4, we have proved the NP-hardness of several versions of the MinDRGT problem. In this section, we consider the problem in a special case where $\mathcal{P} \Leftrightarrow \mathcal{G}$, i.e each gene is the image of a single protein by the mapping function g. In the remaining of the section, we assume that $\mathcal{P} \Leftrightarrow \mathcal{G}$ without explicitly mentioning it. We first study the subcase where $\mathcal{P} \Leftrightarrow \mathcal{G} \Leftrightarrow \mathcal{S}$, i.e each species contains a single gene of the family. Next, we study the case where $\mathcal{P} \Leftrightarrow \mathcal{G}$ and develop a heuristic method for it. In the sequel, given a protein tree P on $\mathcal{P}$, $g(P)$ denotes the gene tree for $\mathcal{G}$ obtained from P by replacing each leaf protein $x \in \mathcal{P}$ by the gene $g(x)$ (see Figure 4 for example). Notice that for any cost function X, $X(P, g(P)) = 0$.

5.1. *Case where $\mathcal{P} \Leftrightarrow \mathcal{G} \Leftrightarrow \mathcal{S}$.*

In this section, we consider the additional restriction that $\mathcal{G} \Leftrightarrow \mathcal{S}$. For a gene tree G on $\mathcal{G}$, $s(G)$ denotes the species tree for $\mathcal{S}$ obtained from G by replacing each leaf gene $x \in \mathcal{G}$ by the species $s(x)$.

One question of interest is whether $g(P)$ is always a solution for MinDRGT$_{XY}$. In other words, is it the case that for any gene tree G', $Y(g(P), S) \leq X(P, G') + Y(G', S)$? When $X = C$ and $Y = D$, this is true if and only if the duplication cost satisfies the triangle inequality. In [22], the authors believed that the duplication cost did have this property, but as we show in Figure 4, this is not always the case. In fact, $g(P)$ cannot be assumed to be optimal also for the case $X = Y \in \{L, M\}$. Thus we get the following remark.

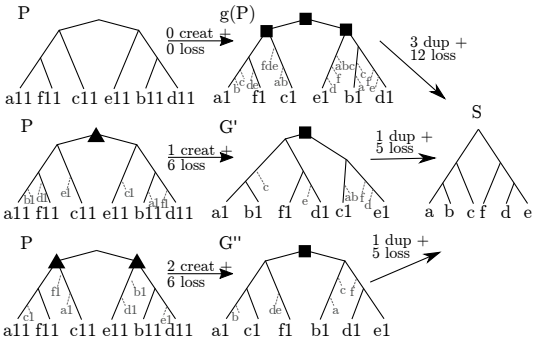


Fig. 4: Example of protein tree P on $\mathcal{P} = \{a_{11}, b_{11}, c_{11}, d_{11}, e_{11}, f_{11}\}$, species tree S on $\mathcal{S} = \{a, b, c, d, e, f\}$ and gene family $\mathcal{G} = \{a_1, b_1, c_1, d_1, e_1, f_1\}$, with $g(x_{ij}) = x_i$ and $s(x_i) = x$ for any protein $x_{ij} \in \mathcal{P}$, gene $x_i \in \mathcal{G}$ and species $x \in \mathcal{S}$. The gene tree G' on $\mathcal{G}$ induces a cost that is strictly lower than the cost induced by the gene tree $g(P)$, for the following double reconciliation costs : CD, CL, CM, LL, LM, MM. The gene tree G'' is the tree that Protein2GeneTree would output.

Remark 1. Figure 4 illustrates that under the restriction that $\mathcal{P} \Leftrightarrow \mathcal{G} \Leftrightarrow \mathcal{S}$, in particular the duplication cost, lost cost and the mutation cost do not satisfy the triangle inequality, i.e., there may exists a gene tree G' on $\mathcal{G}$ such that $D(g(P), S) > C(P, G') + D(G', S)$, $L(g(P), S) > L(g(P), s(G')) + L(G', S)$ and $M(g(P), S) > M(g(P), s(G')) + M(G', S)$. Moreover, the gene tree $g(P)$ is not a solution for any of MinDRGT$_{CD}$, MinDRGT$_{LL}$, MinDRGT$_{CL}$, MinDRGT$_{CM}$, MinDRGT$_{LL}$, MinDRGT$_{LM}$, and MinDRGT$_{MM}$.

5.2. *Case where $\mathcal{P} \Leftrightarrow \mathcal{G}$.*

In the following, we present a heuristic method for the MinDRGT$_{XY}$ problem, under the restriction that $|\mathcal{P}| = |\mathcal{G}|$. The intuition behind the algorithm is based on the idea that we seek for a gene tree G_{opt} on $\mathcal{G}$ that decreases the reconciliation cost with S, while slightly increasing the reconciliation cost with P, in order to globally decreases the double reconciliation cost with P and S. Let $G = g(P)$. The method consists in building G_{opt} from G by slightly modifying subtrees of G that are incongruent with S, in order to decrease the reconciliation cost with S. The heuristic method makes the following choices:
C1) the subtrees of G incongruent with S are those rooted at duplication nodes x with $l_G(x) \neq l_G(x_l)$ or $l_G(x) \neq l_G(x_r)$.
C2) If $l_G(x) = Dup$ and $l_G(x) \neq l_G(x_l)$ w.l.o.g, we denote by $Mix(G[x])$ the set of trees G' on $\mathcal{L}(G[x])$ that can be obtained by grafting $G[x_l]$ onto an edge of $G[x_r]$ on which $s(x_l)$ is lost. Then, the slight modification applied on $G[x]$ consists in replacing $G[x]$ by a tree $G' \in Mix(G[x])$ that decreases the double reconciliation cost with P and S by at least 1.

Protein2GeneTree: Heuristic for MinDRGT$_{XY}$
Input : Tree P on $\mathcal{P}$, Tree S on $\mathcal{S}$, gene set $\mathcal{G}$, mappings g, s.
Output : Tree G_{opt} on $\mathcal{G}$ such that $G_{opt} = g(P)$ or $XY(P, G_{opt}, S) < XY(P, g(P), S)$
1) $G \leftarrow g(P)$
2) Compute l_G and let $\mathcal{D} = \{x \in \mathcal{V}(G) \mid l_G(x) = Dup \text{ and } l_G(x) \neq l_G(x_l) \text{ or } l_G(x) \neq l_G(x_r)\}$
3) For any node $x \in \mathcal{D}$:
   a) $u \leftarrow$ the single node u of P s.t. $g(u) = x$;
   b) $v \leftarrow$ the single node v of S s.t. $v = s(x)$;
   c) $G_{opt}[x] \leftarrow argmax_{G' \in Mix(G[x])} XY(P[u], G', S[v])$
   d) $\delta(x) \leftarrow Y(G[x], S[v]) - XY(P[u], G_{opt}[x], S[v])$
4) Find a subset $\mathcal{D}'$ of $\mathcal{D}$ s.t. $\forall x \in \mathcal{D}'$, $\delta(x) > 0$, and $\forall (x, y) \in \mathcal{D}' \times \mathcal{D}'$, $lca(x, y) \neq x$ and $lca(x, y) \neq y$, and $\sum_{x \in \mathcal{D}'} \delta(x)$ is maximized.
5) Build G_{opt} from G by replacing any subtree $G[x], x \in \mathcal{D}'$ by $G_{opt}[x]$.

Complexity: For Step 4 of Protein2GeneTree, we use a linear-time heuristic greedy algorithm. The time complexity of Protein2GeneTree is in $O(n^2)$ where $n = |\mathcal{G}|$,

since $|\mathcal{V}(G)| = O(n)$, $|\mathcal{D}| = O(n)$ and $|Mix(G[x])| = O(n)$ for any $x \in \mathcal{D}$, and Steps 4 and 5 are realized in linear-time.

For example, the application of Protein2GeneTree on the example of protein tree P and species tree S depicted in Figure 4 would allow to reconstruct the gene tree G'' obtained by moving the subtree of $g(P)$ containing gene c_1 onto the branch leading to gene a_1, and moving the subtree containing gene e_1 onto the branch leading to gene d_1. However, the resulting gene tree G'' is not as optimal as the gene tree G'. Protein2GeneTree can be extended in order to allow computing the more optimal gene tree G' by modifying the choices C1 and C2 made by the algorithm: for example, in Step 2 set $\mathcal{D} = \{x \in V(G) \mid l_G(x) = Dup\}$, and in Step 3.c consider $Mix(G[x]) = \{G' \mid G'|_{\mathcal{L}(G[x_l])} = G[x_l] \text{ and } G|_{\mathcal{L}(G[x_r])} = G[x_r]\}$. The resulting algorithm would be an exponential time algorithm because of the exponential size of the sets $Mix(G[x])$.

6. MinDRPGT for maximum creation-free protein subtrees

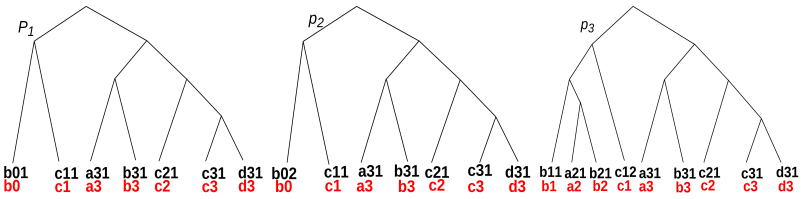
In this section, we consider the MinDRPGT problem in a special case where the input subtrees $P_{i,1\leq i\leq k}$ are all the inclusion-wise maximum creation-free protein subtrees of the real protein tree. For example, the inclusion-wise maximum creation-free protein subtrees of the labeled protein tree P depicted in Figure 2 are the subtrees P_1, P_2, P_3 of P induced by the subsets of proteins $\mathcal{P}_1 = \{b_{01}, c_{11}, a_{31}, b_{31}, c_{21}, c_{31}, d_{31}\}$, $\mathcal{P}_2 = \{b_{02}, c_{11}, a_{31}, b_{31}, c_{21}, c_{31}, d_{31}\}$, and $\mathcal{P}_3 = \{b_{11}, a_{21}, b_{21}, c_{12}, a_{31}, b_{31}, c_{21}, c_{31}, d_{31}\}$. We develop an exact algorithm for reconstructing the protein supertree P given the input subtrees $P_{i,1\leq i\leq k}$, which allows to reduce MinDRPGT to a special case of MinDRGT. The intuition behind the algorithm is to first reconstruct a partition of P into subtrees that are either entirely included or excluded in each input subtree P_i and then combine these partition subtrees into P. The following describes how the partition subtrees of P called *span partition* subtrees are inferred from the input maximum creation-free protein subtrees.

Let P be a protein tree for $\mathcal{P}$ with a LCA-reconciliation l_P, and $\mathbb{P} = \{P_1, P_2, \ldots, P_k\}$ the set of all the inclusion-wise maximum creation-free protein subtrees of P. We define the function *span* from the set of protein $\mathcal{P}$ to the set $2^{\mathbb{P}}$ of subsets of $\mathbb{P}$ such that, for any $x \in \mathcal{P}$, $span(x)$ is the subset of $\mathbb{P}$ such that x is a leaf of any tree in $span(x)$, and x is not a leaf of any tree in $\mathbb{P} - span(x)$. For example, for Figure 2, $span(b_{01}) = \{P_1\}$, $span(c_{11}) = \{P_1, P_2\}$, $span(b_{11}) = \{P_3\}$, $span(a_{31}) = \{P_1, P_2, P_3\}$.

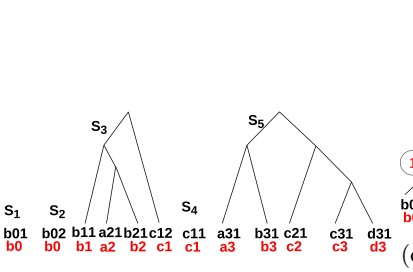
We define the *span partition* of $\mathcal{P}$ according to $\mathbb{P}$ as the partition $\mathbb{P}_{span} = \{S_1, S_2, \ldots, S_m\}$ of $\mathcal{P}$ such that for any set $S_u \in \mathbb{P}_{span}$, for any pair of proteins x, y in S_u, $span(x) = span(y)$. Note that $\mathbb{P}_{span}$ is unique. The function *span* is extended to be defined from $\mathcal{P} \cup \mathbb{P}_{span}$ to $2^{\mathbb{P}}$ such that for $S_u \in \mathbb{P}_{span}$, $span(S_u) = span(x)$ for any $x \in S_u$. The function g is extended to be defined from $\mathcal{P} \cup \mathbb{P}_{span}$ to $2^{\mathcal{G}}$ such that for $S_u \in \mathbb{P}_{span}$, $g(S_u)$ is the set of gene of proteins that belong to S_u.

For example, Figure 5 gives an illustration of the construction of the set of span

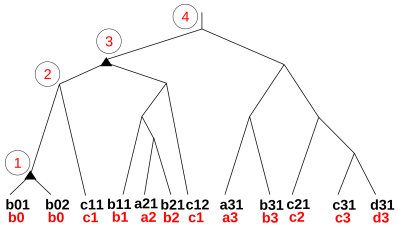
partition subtrees from an input set of maximum creation-free protein subtrees. Say we have the input set of maximum creation-free protein subtrees $\mathbb{P} = \{P_1, P_2, P_3\}$ (Figure 5a). The span partition of $\mathcal{P}$ according to $\mathbb{P}$ is $\mathbb{P}_{span} = \{S_1 = \{b_{01}\}, S_2 = \{b_{02}\}, S_3 = \{b_{11}, a_{21}, b_{21}, c_{12}\}, S_4 = \{c_{11}\}, S_5 = \{a_{31}, b_{31}, c_{21}, c_{31}, d_{31}\}\}$, and $span(S_1) = \{P_1\}$, $span(S_2) = \{P_2\}$, $span(S_3) = \{P_3\}$, $span(S_4) = \{P_1, P_2\}$ and $span(S_5) = \{P_1, P_2, P_3\}$. The set of span partition subtrees is then obtained by building a tree for each element of the span partition (Figure 5b).



(a) Input set of maximum creation-free protein subtrees $\mathbb{P} = \{P_1, P_2, P_3\}$



(b) Span partition trees



(c) Output partially labeled protein supertree

Fig. 5: Illustration of the steps of the algorithm ProteinSuperTree on (a) an input set of maximum creation-free protein subtrees $\mathbb{P} = \{P_1, P_2, P_3\}$. (b) The span partition of $\mathcal{P}$ according to $\mathbb{P}$ contains 5 sets. (c) The output supertree P with numbers in circles representing the order of span partition subtrees merging in the algorithm.

The following lemma describes a property of a set of span partition subtrees that allows to reconstruct a super protein tree from its set of span partition subtrees.

Lemma 1. *Let $\mathbb{P}$ be the set of all maximum creation-free protein subtrees of a labeled protein tree P and $\mathbb{P}_{span}$ be the span partition of $\mathcal{P}$ according to $\mathbb{P}$.*

If P contains at least one protein creation node, then there exist at least a pair of distinct sets S_u, S_v in $\mathbb{P}_{span}$ such that the subtrees of P induced by S_u and S_v, $P|_{S_u}$ and $P|_{S_v}$, are complete subtrees of P, and the subtree of P induced by $S_u \cup S_v$ is also a complete subtree of P. In this case:
(1) $l_P(lca_P(S_u \cup S_v)) = Creat$;
(2) For any $t \in \{u, v\}$ and any $P_i \in span(S_t)$, $P|_{S_t} = P_i|_{S_t}$;
(3) $span(S_u) \cap span(S_v) = \emptyset$;

(4) the following two sets of subtrees are equal: $\{P_i|_{\mathcal{L}(P_i)-S_u} \mid P_i \in span(S_u)\} = \{P_i|_{\mathcal{L}(P_i)-S_v} \mid P_i \in span(S_v)\}$.

Proof. There must exist a node w in P such that $l_P(w) = Creat$ and no node $x \neq w$ in $P[w]$ satisfies $l_P(x) = Creat$. Then, $S_u = \mathcal{L}(P[w_l])$ and $S_v = \mathcal{L}(P[w_r])$. □

For any set $S_t \in \mathbb{P}_{span}$, $tree(S_t)$ denotes the (possibly partial) subtree of P such that, for any $P_i \in span(S_t)$, $tree(S_t) = P_i|_{S_t}$.

SuperProteinTree: for reconstructing P from the input set $\mathbb{P}$
Input : Set $\mathbb{P}$ of all inclusion-wise maximum creation-free protein subtrees of a target protein tree P on $\mathcal{P}$, gene set $\mathcal{G}$, mappings g
Output : Protein tree P.
1) Compute the span partition, $Q \leftarrow \mathbb{P}_{span} = \{S_1, \ldots, S_m\}$;
2) Set $\{tree(S_u) \mid S_u \in \mathbb{P}_{span}\}$ as subtrees of P ;
3) While $|Q| > 1$:
a) Compute the set C of pairs of distinct elements (S_u, S_v) of Q^2 such that $span(S_u) \cap span(S_v) = \emptyset$ and $\{P_i|_{\mathcal{L}(P_i)-S_u} \mid P_i \in span(S_u)\} = \{P_i|_{\mathcal{L}(P_i)-S_v} \mid P_i \in span(S_v)\}$.
b) If $C \neq \emptyset$:
i) Pick a pair (S_u, S_v) in C such that S_u and S_v have the highest mean similarity score between their protein sets;
ii) Add a node w in P such that $tree(S_u)$, $tree(S_v)$ become the left and right subtrees of w, resulting in a subtree P';
iii) Set $l_P(w) \leftarrow Creat$ and for any $S_t \in Q$, $span(S_t) \leftarrow span(S_t) - span(S_v)$;
c) Otherwise,
i) Compute the set of pair of distinct elements (S_u, S_v) of Q^2 such that $P|_{S_u}$ was built at a previous iteration of Step 3 and $span(S_u) = span(S_v)$ and $g(S_u) \cap g(S_v) = \emptyset$;
ii) Pick the pair (S_u, S_v) such that S_u and S_v have the highest mean similarity score between their protein sets;
iii) Graft $P|_{S_u}$ onto $P|_{S_v}$ as the sibling of the node of $P|_{S_v}$ such that the resulting tree P' on $S_u \cup S_v$ is compatible with all subtrees $P_i \in span(S_v)$, i.e. $P'|_{S_t} = P_i|_{S_t}$ with $S_t = (S_u \cup S_v) \cap \mathcal{L}(P_i)$;
d) Set $S_w \leftarrow S_u \cup S_v$ and $Q \leftarrow Q - \{S_u, S_v\} \cup \{S_w\}$ with $span(S_w) \leftarrow span(S_u)$;

Steps 1) and 2) of the algorithm that consist in the computation of the span partition subtrees are illustrated in Figures 5a and 5b . Step 3) that iteratively merges the span partition subtrees in order to obtain the partially labeled protein supertree P is depicted in Figure 5c.

Theorem 2. *Given the set $\mathbb{P}$ of all inclusion-wise maximum creation-free protein subtrees of a labeled protein tree P on $\mathcal{P}$, SuperProteinTree reconstructs P and its time complexity is in $O(n^3)$.*

Proof. The proof follows from the two points below:
(1) For any $S_u \in \mathbb{P}_{span}$, $tree(S_u)$ is a (possibly partial) subtree of P ;
(2) At the end of each iteration of Loop 3, ($Invariant_1$) the subtree P' constructed is a complete subtree of P, and ($Invariant_2$) for any set S_u in Q, there is at most one set $S_v \neq S_u$ in Q such that $span(S_u) = span(S_v)$:

(i) At the first iteration, Q necessarily satisfies case (3.b), and by Lemma 1, the subtree $P' = P|_{S_u \cup S_v}$ must be a complete subtree of P. Moreover, at the end of the iteration, $Invariant_2$ is satisfied.

(ii) Now, let $k \geq 2$. We show that if at the end of each of the first $k-1$ iterations of Loop 3, $Invariant_1$ and $Invariant_2$ were satisfied, then, they are also satisfied at the end of the k^{th} iteration. At the k^{th} iteration:

• If we are in case (3.b) of the algorithm, then there exist two distinct sets S_u, S_v in Q such that $span(S_u) \cap span(S_v) = \emptyset$ and $\{P_i|_{\mathcal{L}(P_i)-S_u} \mid P_i \in span(S_u)\}$ $= \{P_i|_{\mathcal{L}(P_i)-S_v} \mid P_i \in span(S_v)\}$. If S_u and S_v were both elements of the initial partition $Q = \mathbb{P}_{span}$, then from Lemma 1, $P' = P|_{S_u \cup S_v}$ is a complete subtree of P . If one of them, say S_u was not an element of $\mathbb{P}_{span}$, then $P|_{S_u}$ is a complete subtree of P by hypothesis.

• If we are in case (3.c) i.e $span(S_u) = span(S_v)$, then S_u and S_v can not be both elements of $\mathbb{P}_{span}$. So, by hypothesis, one of them, say S_u, is such that $P|_{S_u}$ is a complete subtree.

• So, in both cases (3.b) and (3.c), either $P' = P|_{S_u \cup S_v}$ is a complete subtree of P, or one of the two merged subtrees, say $P|_{S_u}$, is a complete subtree of P. In the later case, suppose that $P|_{S_u \cup S_v}$ is not a complete subtree of P. Then, there exists a protein $x \in \mathcal{P} - \{S_u \cup S_v\}$ such that $x \in \mathcal{L}(P|_{S_v})$ but $x \notin \mathcal{L}(P|_{S_u})$. Let $S_t \in Q$ be the set such that $x \in S_t$. We have $span(S_u) \subseteq span(S_t) \subseteq span(S_v)$, and then $span(S_u) = span(S_t) = span(S_v)$. So, at the end of the $(k-1)^{th}$ iteration, we had $span(S_t) = span(S_u) = span(S_v)$, which is impossible.

• At the end of the k^{th} iteration, $Invariant_2$ is satisfied, otherwise there were two sets S_t, S_q in Q at the end of the $(k-1)^{th}$ iteration such that $span(S_t) = span(S_q)$, which is impossible.

The time complexity of SuperProteinTree is in $O(n^3)$ since the loop at Step 3 is in $O(n)$ and Steps 3.a, 3.b and 3.c are in $O(n^2)$. □

Applying the algorithm SuperProteinTree on an instance of MinDRPGT such that the input subtrees $P_{i,1 \leq i \leq k}$ are all the inclusion-wise maximum creation-free protein subtrees of the real protein tree P, allows to reconstruct P with a partial labeling l_P indicating all protein creation nodes. Then, MinDRPGT is reduced to MinDRGT in the special case where a partial labeling of the input protein tree is given.

7. Application

7.1. *Protein2GeneTree*

We applied the algorithm Protein2GeneTree for the reconstruction of gene trees using protein trees and gene families of the Ensembl database release 87 [16]. Some of the trees were left unchanged by the algorithm. We call an Ensembl gene tree G *modified* if Protein2GeneTree, when given G, outputs a different tree. Otherwise we say that G is *unmodified*. The results are summarized in Table 1. They show that initial gene trees, and particularly large size trees, are predominantly suboptimal in terms of double reconciliation cost. Moreover, modified and unmodified trees have comparable numbers of duplications, but modified trees have significantly higher number of losses, suggesting that gene trees with many losses are susceptible to correction.

Table 1: Results of Protein2GeneTree on 12680 Ensembl gene trees. Samples: (A) $1 \leq n \leq 9$ (7500 trees), (B) $10 \leq n \leq 99$ (4386 trees), (C) $100 \leq n \leq 199$ (773 trees), where n is the number of leaves in a tree with the number of trees in each sample in parenthesis (1) Number and percentage of modified trees, (2) Average number of duplications / losses in unmodified trees, (3) Average number of duplications / losses in modified trees (before modification), (4) Average value / percentage of double reconciliation cost reduction on modified trees, (5) Average running time in ms, (6) Percentage of Ensembl trees that pass the AU test / percentage of corrected trees that pass the AU test, with the percentage of corrected trees with a better AU value in parentheses.

	(1)	(2)	(3)	(4)	(5)	(6)
(A)	72 / 0.96%	0.88/ 7.97	1.61/ 36.61	10.18/ 19.69%	15	93.2% / 71.2% (39.0%)
(B)	1734/ 39.53%	3.83/ 25.04	11.26/ 114.95	7.17/ 5.82%	328	80.6% / 81.1% (45.9%)
(C)	505/ 65.32%	31.54/ 168.8	31.38/ 310.62	16.42/ 4.42%	4685	81.7% / 80.4% (51.1%)

In order to demonstrate the relevance of the gene tree correction achieved using Protein2GeneTree, the corrected trees were evaluated using the AU (Approximately Unbiased) test [28]. This test compares the likelihood of a set of trees in order to build a confidence set, which consists of the trees that cannot be statistically rejected as a valid hypothesis. We say a tree can be rejected if its AU value, which is interpreted as a p-value, is under 0.05. Otherwise, no significant evidence allows us to reject one of the two trees. We start by executing PhyML [14] on the trees to obtain the log-likelihood values per site and we execute Consel [29] to obtain the results from the AU test.

Column 6 of Table 1 shows the percentage of the Ensembl trees/corrected trees that could not be rejected. The corrected trees were in the confidence set about as often as Ensembl, with the exception of the trees with less than 10 leaves. In total, 80.7% of the corrected trees were part of the confidence set and could not be statistically rejected. As for the Ensembl gene trees, 81.2% of them were part of the confidence set. Therefore, despite the fact that our correction algorithm does not consider the likelihood, the majority of the corrected trees represent a hypothesis that cannot be rejected on a statistical basis. Moreover, the corrections are significantly better than the original trees as often as the originals are better than the correction. For the scores themselves, in total there were 52.8% of the Ensembl gene tree that obtained a better AU score than the corrected trees, versus 46.8% of the corrected trees having a better AU score (the remaining 0.4% of the trees having equal value).

7.2. *SuperProteinTree*

We applied the algorithm SuperProteinTree on two sets of homologous genes from the Ensembl-Compara database release 89 [8] augmented with simulated data. For each gene family, we wanted to reconstruct a putative set $\mathbb{P}$ of inclusion-wise maximum creation-free protein subtrees, and build the protein supertree P from $\mathbb{P}$.

Dataset: The dataset contains 14 genes with their CDS sequences from two gene families, FAM86 and TP63, 7 genes per family, 25 CDS for FAM86 and 72 for TP63. For each gene family, the genes are from six different amniote species which are *human, chimpanzee, mouse, rat, cow and chicken*, including two paralogous genes for *human*. For each gene family, the set of CDS of each gene is augmented with simulated CDS as follows. Given a gene G and any CDS C from another gene in the same gene family, the spliced alignment of the CDS C on the gene G is computed using the spliced alignment tool SPlign [18]. If all exons composing C are aligned onto G, the regions of G aligned with exons of C are excised and joined together to simulate an alternative splicing resulting in a CDS C_G of G that is orthologous to the CDS C. If the CDS C_G is not already an existing CDS of the gene G, it is added to its set of CDS as a simulated CDS. By applying this procedure on each pair of CDS and gene in a gene family, we obtain an augmented dataset composed of the same 7 genes per family with their initial sets of CDS augmented with additional CDS, simulated using existing CDS of other genes. Table 2 gives more details about the dataset.

Moreover, the augmented dataset of CDS is partitioned into a set of hard clusters representing protein orthology groups based on the orthology relations inferred using the SPlign spliced alignments. For the the gene family FAM86, 8 orthology groups were inferred, numbered from Group 0 to Group 7, and for the gene family TP63, 24 groups were inferred, numbered from Group 0 to Group 23. The name of each CDS in Figures 6 and 7 is prefixed with the putative orthology group and the

gene to which it belongs.

Table 2: Detailed description of the dataset for the application of the algorithm ProteinSuperTree.

	Family			
	FAM86		**TP63**	
Species	**Gene ID**	**#CDS**	**Gene ID**	**#CDS**
Human	ENSG00000158483	3 ; 0	ENSG00000073282	11 ; 3
Human	ENSG00000186523	4 ; 3	ENSG00000078900	9 ; 0
Chimpanzee	ENSPTRG00000007738	1 ; 1	ENSPTRG00000015733	1 ; 9
Mouse	ENSMUSG00000022544	1 ; 2	ENSMUSG00000022510	8 ; 2
Rat	ENSRNOG00000002876	1 ; 3	ENSRNOG00000001924	5 ; 4
Cow	ENSBTAG00000008222	1 ; 3	ENSBTAG00000015460	1 ; 9
Chiken	ENSGALG00000002044	1 ; 1	ENSGALG00000007324	2 ; 8
Total		12 ; 13		37 ; 35

For each gene family, the family identifier is given together with, for each gene, the species, the Ensembl identifier, the number of CDS (initial CDS ; simulated CDS) of the genes.

Soft-clustering and construction of a putative set of maximum creation-free protein subtrees: Before applying SuperProteinTree, the set $\mathbb{P}$ of all inclusion-wise maximum creation-free protein subtrees of P should be computed. As the target tree P is not given as input, we designed a heuristic method for computing the putative set of subtrees $\mathbb{P}$. The first step of the method consists in soft-clustering the proteins into ortholog groups. The second step consists in building a subtree for each of the ortholog groups.

The soft-clustering step uses a distance measure d that associates to a pair of protein $(x, y) \in \mathcal{P}$ a distance $d(x, y)$ that is obtained from a linear combination of the pairwise similarity score of their nucleotide sequence alignment and a structural similarity score. The multiple alignment of all the CDS of a gene family is computed using the coding sequence alignment tool MACSE [26], and all pairwise alignments between proteins are extracted from the multiple alignment. The structural similarity is evaluated as the number of pairs of exons in the two proteins that are aligned together divided by the maximum number of exons in the two proteins. For each gene family, we used different pairs of values (α,β) for the weights associated to the structural similarity and the sequence similarity in the linear combination for the definition of the distance measure d ((1.0,0.0), (0.8,0.2), (0.6,0.4), (0.4,0.6), (0.2,0.8), and (0.0,1.0)). For each pair of weights for the linear combination, the distance measure d was normalized to a range between 0 and 1.

For the soft-clustering step, the first aim is to allow a protein to appear

in more than one cluster. For example, for the protein tree depicted in Figure 2, the target ortholog soft-clusters are the maximum creation-free subtrees $\mathcal{P}_1 = \{b_{01}, c_{11}, a_{31}, b_{31}, c_{21}, c_{31}, d_{31}\}$, $\mathcal{P}_2 = \{b_{02}, c_{11}, a_{31}, b_{31}, c_{21}, c_{31}, d_{31}\}$, and $\mathcal{P}_3 = \{b_{11}, a_{21}, b_{21}, c_{12}, a_{31}, b_{31}, c_{21}, c_{31}, d_{31}\}$. The second aim of the soft-clustering is to prevent a cluster to contain more than one protein from each gene, as in such a case two proteins of a same gene in a cluster would be paralogs.

Our clustering algorithm is the UPGMA (Unweighted Pair Group Method with Arithmetic Mean) hierarchical clustering algorithm modified as follows. (1) When a new cluster K resulting from the merging of two minimum-distance clusters I and J is added, the clusters I and J are not discarded. They remain candidates for future merging with other clusters. (2) Two clusters containing proteins from the same gene cannot be merged. (3) At each step, the distance between a new cluster K and a cluster L is computed as $D(K, L) = \max\{d(x, y) \mid (x, y) \in K \times L\}$.

For each gene family and for each pairs of values (α,β) for the weights of the structural similarity and the sequence similarity in the definition of the distance measure d, we applied the soft-clustering algorithm and outputed different sets of soft clusters obtained obtained after different numbers γ of iterations in the clustering ($n/2, n, 2n$ and $3n$ where n is the number of proteins in $\mathcal{P}$). Each value of the number of iterations γ in the clustering then resulted in a set of soft clusters for the set of protein P. For a given set of soft clusters obtained for a set of parameters (α,β, γ), the corresponding set of input subtrees for the SuperProteinTree algorithm was computed as follows. PhyML [14] was used to build an initial tree P' for the set of all proteins P. For each ortholog group K in the set of soft clusters, the subtree of P for K was then defined as the subtree of P' induced by K.

Results of applying the SuperProteinTree algorithm: For each gene family and each set of input subtrees obtained for different values of the parameters α, β, and γ in the hierarchical soft clustering, we applied the algorithm SuperProteinTree to reconstruct a protein supertree. We then retained the best reconstructed protein supertree, i.e one that consists in a single tree (not a forest composed of incompatible subtrees) with the minimum number of creation events.

FAM86: Table 3 summarizes the results for the gene family FAM86. Figure 6 shows the initial protein tree P' computed using PhyML [14] and the protein supertree P reconstructed using SuperProteinTree for the best combination of parameter values (given in Table 3), $\alpha = 0.8$, $\beta = 0.2$ and $\gamma = n = 25$. The tree P' contains 18 proteins creation events while the supertree P contains 8 proteins creation events.

TP63: Table 4 summarizes the results for the gene family TP63. Figure 7 shows the initial protein tree P' computed using PhyML and the protein supertree P reconstructed using SuperProteinTree for the best combination of parameter values, $\alpha = 0.8$, $\beta = 0.2$, and $\gamma = n = 72$. The tree P' contains 62 proteins creation events while the supertree P contains 28 proteins creation events.

Table 3: SuperProteinTree results obtained on $n = 25$ proteins for the gene family FAM86 for varying values of the parameters α (weight of the structure similarity in the distance measure d), β (weight of the sequence similarity), and γ (number of iteration of the hierarchical soft clustering). For each of set of values, three results **A, B, C** are given: **A** is a binary value that indicates if SuperProteinTree has reconstructed a single protein tree (1) or a forest (0); **B** is the number of creation events inferred in the reconstruction and **C** is the ratio of the number of recovered initial orthology groups (8 initial orthology groups for FAM86). The best results are indicated in bold font.

		γ											
		n/2			**n**			**2n**			**3n**		
α	β	**A**	**B**	**C**	**A**	**B**	**C**	**A**	**B**	**C**	**A**	**B**	**C**
1.0	**0.0**	1	17	1/8	1	18	2/8	0	12	2/8	0	7	1/8
0.8	**0.2**	1	15	3/8	**1**	**8**	**6/8**	0	10	6/8	0	9	3/8
0.6	**0.4**	1	17	1/8	1	16	1/8	0	9	3/8	0	8	3/8
0.4	**0.6**	1	17	1/8	0	15	1/8	0	12	5/8	0	6	1/8
0.2	**0.8**	0	16	1/8	1	17	1/8	0	13	3/8	0	7	1/8
0.0	**1.0**	1	18	1/8	0	15	1/8	1	16	2/8	0	14	1/8

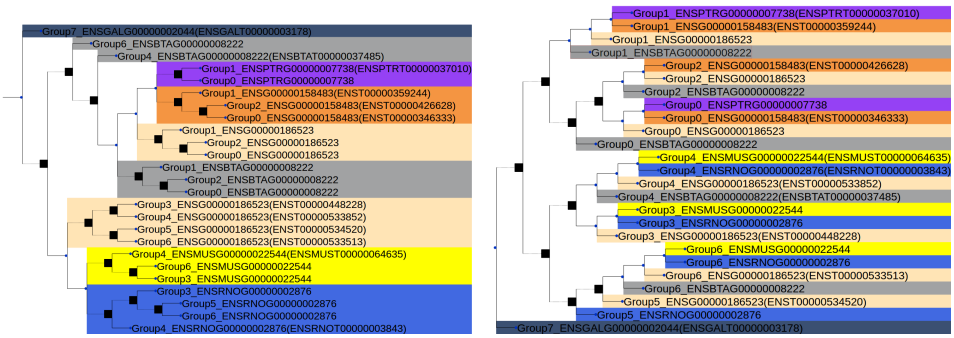


Fig. 6: **Left.** The initial protein tree P' computed using PhyML. Protein lines are colored according to the gene to which they belong (blue: *ENSRNOG00000002876* ; dark blue : *ENSGALG00000002044* ; orange : *ENSG00000158483* ; gray : *ENSBTAG00000008222* ; beige : *ENSG00000186523* ; yellow : *ENSMUSG00000022544*); purple : *ENSPTRG00000007738*. **Right.** The protein supertree P reconstructed using SuperProteinTree. The creation events inferred in the protein trees after reconciliation with the gene tree are represented as square nodes in black colors.

Discussion: As expected, for both gene families, FAM86 and TP63, in the protein tree P' computed using a sequence-based reconstruction method, the proteins of each gene are grouped into complete subtrees. As a consequence, all the creation

Table 4: SuperProteinTree results obtained on $n = 72$ proteins for the gene family TP63. The notation is the same as the one used for Table 3, except that the dataset for TP63 contains 24 initial orthology groups instead of 8 for FAM86.

		γ											
		n/2			**n**			**2n**			**3n**		
α	β	**A**	**B**	**C**	**A**	**B**	**C**	**A**	**B**	**C**	**A**	**B**	**C**
1.0	**0.0**	1	61	14/23	1	62	14/24	1	65	14/24	0	28	14/24
0.8	**0.2**	0	61	14/24	**1**	**28**	**22/24**	0	25	21/24	0	20	19/24
0.6	**0.4**	0	58	14/24	0	51	14/24	0	35	18/24	0	35	18/24
0.4	**0.6**	0	59	14/24	0	45	15/24	0	37	17/24	0	37	17/24
0.2	**0.8**	0	57	14/24	0	52	14/24	0	52	14/24	0	59	14/24
0.0	**1.0**	0	59	14/24	0	58	14/24	0	58	14/24	0	56	14/24

events are located in these subtrees at the bottom of the tree P' and each gene has its specific set of protein creation events not shared by any other gene. However in the protein supertree P reconstructed using our method, the creation events are distributed from the root to the leaves of the tree. This supports the hypothesis of groups of orthologous protein isoforms shared by several extant genes, and originated from ancestral protein creation events.

The results given in Table 3 and 4 show that by giving more weight to structural similarity than to sequence similarity between proteins, SuperProteinTree achieved a better performance in both experiments. Specially, when $\alpha = 0.8$ and $\beta = 0.2$, our method is able to reconstruct a single protein tree displaying the highest ratio of recovered initial orthology groups, with the smallest number of creation events. It is worth noting that in the special case when $\alpha = 1.0$ and $\beta = 0.0$, the SuperProteinTree algorithm is able to reconstruct one protein tree, but with more creation events and less recovered initial orthology groups. This might be due to the fact that the SuperProteinTree algorithm tries to group proteins which have the same structure but are too different in terms of sequence. A similar phenomenon occurs when $\alpha = 0.0$ and $\beta = 1.0$. Again, the number of creations is high and the number of recovered initial orthology groups is low, suggesting that the structure of proteins should not be ignored. These results illustrate the advantage of incorporating both pieces of information into the calculation of distance between proteins.

8. Conclusion

In this work, we have argued the importance of distinguishing gene trees from protein trees, and introduced the notion of protein trees into the framework of reconciliation. We have shown that, just as gene trees are thought of as evolving "inside" a species tree, protein trees evolve "inside" a gene tree, leading to two layers of reconciliation. We provided evidence that, even if each gene in a given

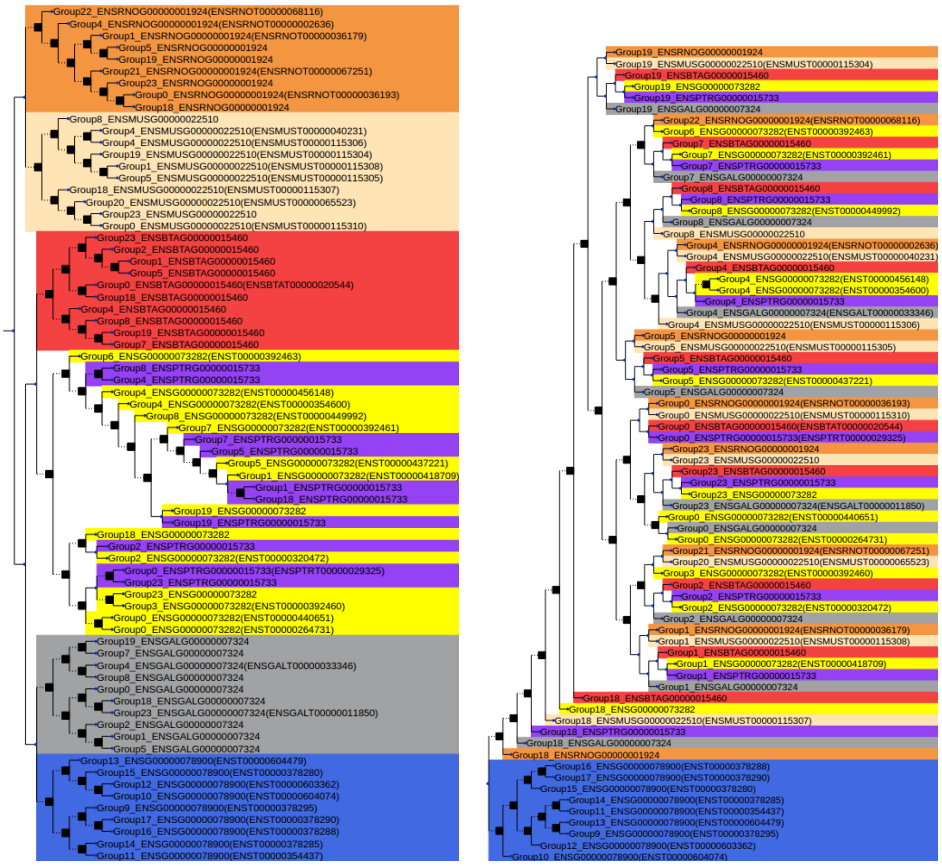


Fig. 7: **Left.** The initial protein tree P' computed using PhyML. Protein lines are colored according to the gene to which they belong.**Right.** The protein supertree P reconstructed using SuperProteinTree. The creation events inferred in the protein trees after reconciliation with the gene tree are represented as square nodes in black colors.

family encodes a single protein, the gene phylogeny does not have to be the same as the protein phylogeny, and may rather behave like a "median" between the protein tree and the species tree in terms of mutation cost. We have also introduced the idea of reconstructing phylogenies from a set of orthology constraints represented as proteins orthology soft-clusters. We have designed an algorithm to reconstruct a protein phylogeny from a set of orthology soft-clusters.

On the algorithmic side, many questions related to the double-reconciliation cost deserve further investigation. For instance, what is the complexity of finding an optimal gene tree in the case that $\mathcal{P} \Leftrightarrow \mathcal{G} \Leftrightarrow \mathcal{S}$? Also, given that the general MinDRGT problem is NP-hard, can the optimal gene tree G be approximated

within some constant factor? Or is the problem fixed-parameter tractable with respect to some interesting parameter, e.g. the number of apparent creations in the protein tree, or the maximum number of proteins per gene? As for the MinDRPGT problem, it remains to explore how the partially labeled protein trees can be used to infer the gene tree. Moreover, we have studied an ideal case where all maximum creation-free protein subtrees could be inferred perfectly. Future work should consider relaxing this assumption by allowing the input subtrees to have missing or superfluous leaves, or to contain errors. Finally, we have designed an algorithm for building a complete phylogeny satisfying a set of orthology soft-clusters in the special case where all orthology soft-clusters are consistent. We defer to a future work the study of the consistency problem for orthology soft-clusters, and the problems of reconstructing a phylogeny from a set of inconsistent orthology soft-clusters.

Acknowledgments

Esaie Kuitche acknowledges the support of Faculty of Sciences of the Université de Sherbrooke.
Manuel Lafond acknowledges the support of the Natural Sciences and Engineering Research Council (NSERC).
Aïda Ouangraoua acknowledges the support of the Canada Research Chairs (CRC), the NSERC and the Fonds de Recherche du Québec - Nature et Technologies (FRQNT).

References

1. Åkerborg Ö, Sennblad B, Arvestad L, Lagergren J, Simultaneous bayesian gene tree reconstruction and reconciliation analysis, *Proceedings of the National Academy of Sciences* **106**(14):5714–5719, 2009.
2. Åkerborg Ö, Sennblad B, Arvestad L, Lagergren J, Simultaneous bayesian gene tree reconstruction and reconciliation analysis, *Proceedings of the National Academy of Sciences* **106**(14):5714–5719, 2009.
3. Barbosa-Morais NL, Irimia M, Pan Q, Xiong HY, Gueroussov S, Lee LJ, Slobodeniuc V, Kutter C, Watt S, Çolak R, *et al.*, The evolutionary landscape of alternative splicing in vertebrate species, *Science* **338**(6114):1587–1593, 2012.
4. Chang WC, Eulenstein O, Reconciling gene trees with apparent polytomies, *COCOON*, Springer, pp. 235–244, 2006.
5. Chauve C, El-Mabrouk N, New perspectives on gene family evolution: Losses in reconciliation and a link with supertrees., *RECOMB*, Springer, pp. 46–58, 2009.
6. Chen K, Durand D, Farach-Colton M, Notung: a program for dating gene duplications and optimizing gene family trees, *Journal of Computational Biology* **7**(3-4):429–447, 2000.
7. Christinat Y, Moret BM, Inferring transcript phylogenies, *BMC bioinformatics* **13**(9):1, 2012.
8. Cunningham F, Amode MR, Barrell D, *et al.*, Ensembl 2015, *Nucleic Acids Research* **43**(D1):D662–D669, 2015.

9. Doyon JP, Ranwez V, Daubin V, Berry V, Models, algorithms and programs for phylogeny reconciliation, *Briefings in bioinformatics* **12**(5):392–400, 2011.
10. Eulenstein O, Huzurbazar S, Liberles DA, Reconciling phylogenetic trees, *Evolution after gene duplication* pp. 185–206, 2010.
11. Goodman M, Czelusniak J, Moore GW, Romero-Herrera A, Matsuda G, Fitting the gene lineage into its species lineage, a parsimony strategy illustrated by cladograms constructed from globin sequences, *Systematic Biology* **28**(2):132–163, 1979.
12. Gorecki P, Eulenstein O, Tiuryn J, Unrooted tree reconciliation: a unified approach, *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)* **10**(2):522–536, 2013.
13. Górecki P, Tiuryn J, Dls-trees: a model of evolutionary scenarios, *Theoretical computer science* **359**(1):378–399, 2006.
14. Guindon S, Gascuel O, A simple, fast, and accurate algorithm to estimate large phylogenies by maximum likelihood, *Systematic biology* **52**(5):696–704, 2003.
15. Hahn MW, Bias in phylogenetic tree reconciliation methods: implications for vertebrate genome evolution, *Genome biology* **8**(7):1, 2007.
16. Hubbard T, Barker D, Birney E, Cameron G, Chen Y, Clark L, Cox T, Cuff J, Curwen V, Down T, *et al.*, The ensembl genome database project, *Nucleic acids research* **30**(1):38–41, 2002.
17. Irimia M, Rukov JL, Penny D, Roy SW, Functional and evolutionary analysis of alternatively spliced genes is consistent with an early eukaryotic origin of alternative splicing, *BMC Evolutionary Biology* **7**(1):188, 2007.
18. Kapustin Y, Souvorov A, Tatusova T, Lipman D, Splign: algorithms for computing spliced alignments with identification of paralogs, *Biology direct* **3**(1):20, 2008.
19. Keren H, Lev-Maor G, Ast G, Alternative splicing and evolution: diversification, exon definition and function, *Nature Reviews Genetics* **11**(5):345–355, 2010.
20. Lafond M, Chauve C, El-Mabrouk N, Ouangraoua A, Gene tree construction and correction using supertree and reconciliation, *IEEE/ACM Transactions on Computational Biology and Bioinformatics (Proceedings of Asia Pacific Bioinformatics Conference APBC'17)* , 2017, dOI 10.1109/TCBB.2017.2720581.
21. Lafond M, Swenson K, El-Mabrouk N, An optimal reconciliation algorithm for gene trees with polytomies, *Algorithms in Bioinformatics* pp. 106–122, 2012.
22. Ma B, Li M, Zhang L, From gene trees to species trees, *SIAM Journal on Computing* **30**(3):729–752, 2000.
23. Nilsen TW, Graveley BR, Expansion of the eukaryotic proteome by alternative splicing, *Nature* **463**(7280):457–463, 2010.
24. Noutahi E, Semeria M, Lafond M, Seguin J, Boussau B, Guguen L, El-Mabrouk N, Tannier E, Efficient gene tree correction guided by genome evolution, *Plos One* , 2016, to appear.
25. Osório J, Evolutionary genetics: Alternative splicing shapes vertebrate evolution, *Nature Reviews Genetics* **16**(10):565–565, 2015.
26. Ranwez V, Harispe S, Delsuc F, Douzery EJ, MACSE: Multiple Alignment of Coding SEquences accounting for frameshifts and stop codons, *PLoS One* **6**(9):e22594, 2011.
27. Rasmussen MD, Kellis M, A bayesian approach for fast and accurate gene tree reconstruction, *Molecular Biology and Evolution* **28**(1):273–290, 2011.
28. Shimodaira H, An approximately unbiased test of phylogenetic tree selection, *Systematic biology* **51**(3):492–508, 2002.
29. Shimodaira H, Hasegawa M, CONSEL: for assessing the confidence of phylogenetoc tree selection, *Bioinformatics* **17**:1246- 1247, 2001.
30. Vilella AJ, Severin J, Ureta-Vidal A, Heng L, Durbin R, Birney E, Ensemblcompara

genetrees: Complete, duplication-aware phylogenetic trees in vertebrates, *Genome research* **19**(2):327–335, 2009.
31. Wu YC, Rasmussen MD, Bansal MS, Kellis M, Treefix: statistically informed gene tree error correction using species trees, *Systematic biology* **62**(1):110–120, 2013.
32. Zambelli F, Pavesi G, Gissi C, Horner DS, Pesole G, Assessment of orthologous splicing isoforms in human and mouse orthologous genes, *BMC genomics* **11**(1):1, 2010.

Esaie Kuitche received the Master's degree in Computer Engineering from École Polytechnique de Yaoundé, Cameroun in 2016. He is a PhD candidate at the Department of Computer Science of Université de Sherbrooke, Québec, Canada, where he develops models and algorithms for the reconstruction of orthology clusters and phylogenies for protein and gene families. He has a scholarship from the Faculty of Science of Université de Sherbrooke.

Manuel Lafond received the PhD degree from the University of Montreal, Canada in 2016. He is now a NSERC postdoctoral fellow in the Department of Mathematics and Statistics at the University of Ottawa, Canada, where he is developing new models and methods to distinguish orthologous genes from paralogous genes. His research interests mainly reside in the intersection of computational biology, algorithms and graph theory.

Aïda Ouangraoua received the PhD degree in Computer Science from University of Bordeaux, France, in 2007. She did her postdoctoral training at Simon Fraser University and Université du Québec à Montréal, Canada. She was a researcher at INRIA Lille, France from 2009 to 2014. She is currently a professor at the Department of Computer Science of Université de Sherbrooke, Québec, Canada, where she holds the Canada Research Chair in Computational and Biological Complexity. Her research interests include Algorithms and Computational Genomics.