Reconstructing the history of syntenies through Super-Reconciliation

Mattéo Delabre[1], Nadia El-Mabrouk[1], Katharina T. Huber[2], Manuel Lafond[3], Vincent Moulton[2], and Miguel Sautie Castellanos[1]

[1] Département d'informatique (DIRO), Université de Montréal, Québec, Canada
mabrouk@iro.umontreal.ca
[2] School of Computing Sciences, University of East Anglia, Norwich, UK
[3] Department of Computer Science, Université de Sherbrooke, Sherbrooke, Canada

Abstract. Classical gene and species tree reconciliation, used to infer the history of gene gain and loss explaining the evolution of gene families, assumes an independant evolution for each family. While this assumption is reasonable for genes that are far apart in the genome, it is clearly not suited for genes appearing grouped in syntenic blocks, which are more plausibly the result of a concerted evolution. Here, we introduce the *Super-Reconciliation* model extending the Duplication-Loss reconciliation model to the reconciliation of a set of trees, accounting for segmental duplications and losses. From a complexity point of view, we show the associated decision problem is NP-hard. We then give an exact algorithm allowing to solve it, test its time efficiency on simulated datasets, and give a proof of concept on the opioid receptor genes.

Keywords: Gene Tree · Reconciliation · Duplication · Loss · synteny

1 Introduction

Gene gain and loss is known as a major force driving evolution. Assuming the gene and species trees are known and represent the true evolution, incongruence between the two trees can be explained from gene gain and loss events, and "reconciling" the two trees allows recovering these events.

Tree reconciliation can be performed through different biological models of evolution, the most common being the Duplication-Loss (DL) [13, 30, 31] or Duplication-Loss and Transfer [5, 9, 28] models. While most reconciliation methods are based on the parsimony principle of minimizing the number or cost of operations, probabilistic models seeking for a reconciliation with maximum likelihood or maximum posterior probability have also been developed [4, 24, 27].

Whatever the model, current algorithms for reconciliation take each gene family individually, assuming an independent evolution through single duplications and losses. Although this hypothesis holds for genes that are far apart in the genome, it is clearly too restrictive for those organized in syntenic blocks or paralogons, i.e. sets of homologous chromosomal regions, among one or many genomes, sharing the same genes (e.g. neuropeptide Y-family receptors [19], the

Homeobox gene clusters [1, 11, 12], the FGFR fibroblast growth factor receptors [3, 14] or the genes of the opioid system [10, 25, 26]). They are more plausibly the result of an evolution from a common ancestral region, rather than of a set of independent gene duplications that would have converged to the same organization in different genomic regions.

The purpose of this paper is to generalize the DL reconciliation model from a unique gene tree to a set of gene trees, accounting for segmental duplications and losses. As far as we know, this problem has never been considered before. The closest algorithms are DeCo [6] and DeCoStar [29] which, given a set of gene families, a set of adjacencies between genes, a set of gene trees and a species tree, output an adjacency forest reflecting the evolution of each adjacency. However, adjacencies are taken independently, and still only single duplications and losses are considered. Another related work is by Paszek and Górecki [21] asking for the reconciliation of a set of gene trees leading to a minimum number of duplication episodes, defined as sets of single duplications mapped to the same node in the species tree. Again, the model does not account for the possibility of a duplication involving a set of neighboring genes.

We consider the *Super-Reconciliation Problem* such that, given a set of gene families, a set of syntenies, a gene tree for each gene family and a species tree, seeks for an evolutionary history of the set of syntenies, which is in agreement with the individual gene trees, minimizing the number of segmental duplications and losses. After defining the new reconciliation model in the next section, we begin, in Section 3, by characterizing the conditions under which a Super-Reconciliation exists for a set of syntenies and a set of gene trees. We prove, in Section 4, that the Super-Reconciliation problem is NP-hard. We then exhibit a dynamic programming algorithm in Section 5. An application on simulated datasets and a proof of concept on the genes of the opioid system are then presented in Section 6. We conclude with a discussion on future work in Section 7.

2 Trees, Reconciliation and Problem Statement

A *string* or a *sequence* is an ordered set of characters. Given a string $X = x_1 \cdots x_n$, a *substring* of X is a consecutive set of characters from X in the same order as in X, and a *subsequence* is a set of characters of X in the same order, but not necessarily consecutive in X. X is a substring and a subsequence of X.

All trees are considered rooted. Given a tree T, we denote by $r(T)$ its root, by $V(T)$ its set of nodes and by $\mathcal{L}(T) \subset V(T)$ its leafset. We say that T *is a tree for* $L = \mathcal{L}(T)$. A node x is an *ancestor* of y if x is on the path from $r(T)$ to y; x is the *father* of y if it directly precedes y on this path. In this latter case, y is called the *child* of x. We denote by $E(T)$ the set of edges of T, where an edge is represented by its two terminal nodes (x, y), with x being the father of y. Two nodes x and y are *separated* in T iff neither is an ancestor of the other. A node is said to be *unary* if it has a single child and *binary* if it has two children. Given a node x of T, the subtree of T rooted at x is denoted $T[x]$.

A *binary tree* is a tree with all internal (i.e. non-leaf) nodes being binary. If internal nodes have one or two children, then the tree is said *partially binary*.

Creating a unary root consists in creating a new node z, a new edge $(z, r(T))$ and assigning z as the new root of T. *Grafting* a leaf w consists of subdividing an edge (x, y) of T, thereby creating a new node z between x and y, then adding a leaf w with parent z. If W is a rooted tree, *grafting W to T* corresponds to grafting a leaf w, then replacing w by the root of W.

The *lowest common ancestor* (lca) in T of a subset L' of $\mathcal{L}(T)$, denoted $lca_T(L')$, is the ancestor common to all nodes in L' that is the most distant from the root. The restriction $T|_{L'}$ of T to L' is the tree with leafset L' obtained from the subtree of T rooted at $lca_T(L')$ by removing all leaves that are not in L' and all unary nodes. Let T' be a tree such that $\mathcal{L}(T') = L' \subseteq \mathcal{L}(T)$. We say that T *displays* T' iff $T|_{L'}$ is label-isomorphic to T' (i.e. isomorphic with preservation of leaf labels). We also say that T is an *extension* of T'.

Species, gene and synteny trees: (See Figure 1) The *species tree* S for a set Σ of species represents an ordered set of speciation events that have led to Σ.

A *gene family* is a set Γ of genes where each gene g belongs to a given species $s(g)$ of Σ. If $\Gamma' \subseteq \Gamma$ is a subset of genes, we denote $s(\Gamma') = \{s(g) : g \in \Gamma'\}$.

A *synteny* is an ordered sequence of genes. We consider that genes of a synteny all belong to different gene families (tandem duplications are ignored). More precisely, given a set $\mathcal{F} = \{\Gamma_1, \Gamma_2, \cdots, \Gamma_n\}$ of gene families, a synteny is an ordered sequence $X = g_{i_1} \cdots g_{i_k}$ of genes such that, for each $1 \leq j \leq k$, g_{i_j} belongs to the family Γ_{i_j}, the elements in $\{i_j\}_{1 \leq j \leq k}$ are all different and the elements in $\{s(g_{i_j})\}_{1 \leq j \leq k}$ are all equal. If X is a synteny, then $s(X)$ simply denotes the genome containing X. Because a synteny never contains two genes from the same family, we may represent it as a sequence on alphabet $\mathcal{F}$.

A *synteny family* is a set $\mathcal{X}$ of syntenies. We say that a set $\mathcal{F}$ of gene families are *organized into a set $\mathcal{X}$ of syntenies* iff there is a bijection between the genes of $\mathcal{F}$ and the genes in $\mathcal{X}$ (each gene of $\mathcal{F}$ belongs to exactly one synteny of $\mathcal{X}$).

A tree T is a *gene tree* for a gene family Γ (respec. a *synteny tree* for a synteny family $\mathcal{X}$) if its leafset is in bijection with Γ (respec. $\mathcal{X}$).

Given a gene tree T, the *corresponding synteny tree* is the tree $\tilde{T}$ obtained from T by replacing each leaf of T by the synteny containing the considered gene.

Given a tree T (either gene tree or synteny tree), we extend the mapping s to internal nodes x of T by defining $s(x) = lca_S(\{s(l) : l \in \mathcal{L}(T[x])\})$.

An evolutionary history is represented by a *labeled* tree, where the label of a node is its corresponding event. In the case of gene families, an event is fully determined by its type, either a duplication, a speciation or a loss. The labels of a gene tree are obtained through reconcilication, as described below.

2.1 Reconciliation

Definition 1 (Reconciled gene tree). *Let T be a binary gene tree and S be a binary species tree. A* DL reconciliation *(or simply* reconciliation*) $R(T, S)$ of*

T with S is a labeled extension of T obtained by grafting new leaves satisfying: for each internal node x of $R(T,S)$ with two children x_l and x_r, either $s(x_l) = s(x_r) = s(x)$, or $s(x_l)$ and $s(x_r)$ are the two children of $s(x)$. The node x is a duplication in $s(x)$ in the former case and a speciation in the latter case. A grafted leaf on a newly created node x corresponds to a loss in $s(x)$. All other leaves are labeled by the default event "extant".

The cost of a reconciliation $R(T,S)$ is the number of induced duplications and losses.

Given a gene tree T and a species tree S, a *minimum reconciliation*, i.e. a reconciliation of minimum cost, is obtained from the LCA-mapping which consists in setting $s(x) = lca_S(s(\mathcal{L}(T[x])))$ for each $x \in V(T)$, and labeling each internal node x of T as a speciation if and only if $s(x_l)$ and $s(x_r)$ are separated in S, and as a duplication otherwise. Observe that in any case, if $s(x_l)$ and $s(x_r)$ are not separated, then it is impossible for x to be a speciation. We denote by *LCA-reconciliation* the reconciliation labeled by means of the LCA-mapping.

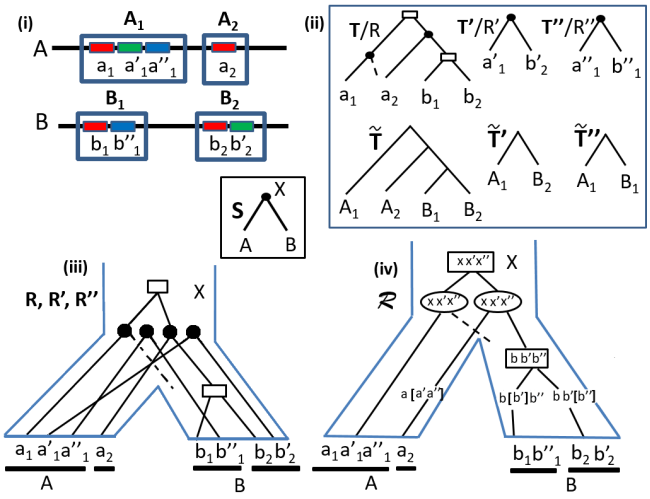

Fig. 1. (i) Two genomes A and B; three gene families (red, green and blue) grouped into two syntenies A_1, A_2 in A and two syntenies B_1, B_2 in B. (ii) Ignoring node labels and dotted lines, T, T' and T'' are the corresponding gene trees and $\tilde{T}$, $\tilde{T}'$ and $\tilde{T}''$ are the corresponding synteny trees. The reconciled gene trees R, R' and R'' are the same trees but including node labels and dotted lines. Nodes identified by circles are speciations, those represented by rectangles are duplications, and dotted lines represent lost branches. (iii) The reconciled trees embedded in the species tree S. (iv) A Super-Reconciliation $\mathcal{R}$, representing a more realistic evolutionary history from a common ancestral synteny. Each ancestral node is identified by the synteny, the event and the segment of the synteny affected by the event. Square nodes represent *Dup* events, round nodes *Spe* events, brackets *pLoss* events and dotted lines *fLoss* (see text).

Before extending the reconciliation concept to a set of gene trees, we need to specify an evolutionary model for syntenies. In this paper, syntenies are consid-

ered to have evolved from a single ancestral synteny through speciation (defined as for single genes), segmental duplication and segmental loss, where:

- a speciation $Spe(X, [1, l])$ acting on a synteny $X = g_1 \cdots g_l$ belonging to a genome $s(X)$ has the effect of reproducing X in the two genomes s_l and s_r children of $s(X)$ in S.
- a (segmental) duplication $Dup(X, [i, j])$ acting on a synteny X belonging to a genome $s(X)$ is an operation that copies a substring $g_i \cdots g_j$ of size $j-i+1$ of $X = g_1 g_2 \cdots g_i \cdots g_j \cdots g_l$ somewhere else into the genome $s(X)$, creating a new *copied synteny* $X' = g'_i \cdots g'_j$ where each g'_k, for $i \leq k \leq j$ belongs to the same gene family as g_k;
- a (segmental) loss $Loss(X, [i, j])$ acting on a synteny $X = g_1 \cdots g_i \cdots g_j \cdots g_l$ is an operation that removes a substring $g_i \cdots g_j$ of size $j-i+1$ of X, leading to the *truncated synteny* $X' = g_1 \cdots g_{i-1} g_{j+1} \cdots g_l$. A loss is called *full* if X' is the empty string (i.e. all genes of X are removed) and *partial* otherwise. We may denote full loss events as $fLoss$ and partial loss events as $pLoss$.

An evolutionary history of a set of syntenies can thus be represented as a partially binary tree where leaves correspond to extant syntenies and lost syntenies (resulting from full losses), and each internal node x corresponds to an event $\mathcal{E}(X, [i, j])$ with $\mathcal{E} \in \{Spe, Dup, pLoss\}$ (and leaves correspond to either extant genes or $fLoss$ events). Thus, in contrast to a single gene family, a tree representing the evolution of a set of syntenies is not only labeled by the type of event corresponding to each internal node, but also by the segment of the synteny affected by the event (see the bottom-right tree in Figure 1). If $\mathcal{E}$ is:

1. Spe, then x is a binary node with two children corresponding to syntenies Y and Z such that $X = Y = Z$ and $s(Y)$ and $s(Z)$ being the two children of $s(X)$ in S.
2. Dup, then x is a binary node with two children corresponding to syntenies X and $X' = X[i, j]$, where $s(X) = s(X')$.
3. $pLoss$, then x is a unary node with a child corresponding to the truncated synteny $X' = X[1, i-1]X[j+1, l]$, and $s(X) = s(X')$.

The topology of a tree representing the evolution of a set of syntenies differs from that of a single gene family since the former may contain unary nodes, resulting from partial losses, while the latter only contains binary nodes.

Our goal is to infer an evolutionary history of a set of syntenies which is a reconciliation of a set of individual gene trees, formally defined below.

Definition 2 (Super-Reconciliation). *Let $\mathcal{G} = \{T_1, T_2, \cdots, T_n\}$ be a set of binary gene trees for the gene families $\mathcal{F} = \{\Gamma_1, \Gamma_2, \cdots, \Gamma_n\}$ organized into a set $\mathcal{X}$ of syntenies belonging to a set Σ of taxa, and let S be a binary species tree for Σ. For each i, $1 \leq i \leq n$, let $\tilde{T}_i$ be the synteny tree corresponding to T_i.*

A Super-Reconciliation *$R(\mathcal{G}, S)$ of $\mathcal{G}$ with S is a labeled synteny tree which is an extension of the trees $\tilde{T}_i$, for $1 \leq i \leq n$, representing a valid history for $\mathcal{X}$.*

The cost of a Super-Reconciliation $R(\mathcal{G}, S)$ is the number of induced Dup, $fLoss$ and $pLoss$ events.

For example, the cost of the Super-Reconciliation in Figure 1 is 6. We are now ready to state the optimization problem considered in this paper.

Super-Reconciliation Problem:
Input: A set Σ of species and a species tree S for Σ; a set of gene families $\mathcal{F} = \{\Gamma_1, \Gamma_2, \cdots, \Gamma_n\}$ organized into a set of syntenies $\mathcal{X}$; a set of gene trees $\mathcal{G} = \{T_1, T_2 \cdots, T_n\}$ one for each family of $\mathcal{F}$;
Output: A Super-Reconciliation $R(\mathcal{G}, S)$ of minimum cost.

3 Existence conditions

As a synteny is represented by a gene order and can only be modified through losses (duplications create new syntenies but do not modify existing syntenies), an evolutionary history does not always exist for a set of syntenies $\mathcal{X}$, regardless of the trees linking them. If this holds, the syntenies are said to be *order consistent*. Due to space constraints, we leave the details on order consistency constaints in the supplementary material.

In addition, in contrast to the reconciliation of a single gene tree which always exists, this is not the case for a Super-Reconciliation as different gene trees may exhibit an inconsistent speciation history for the same syntenies. A set of trees on subsets of $\mathcal{X}$ is said *consistent* iff, for any triplet $Trp = \{X_1, X_2, X_3\}$ of disjoint elements of $\mathcal{X}$, all trees containing Trp as a sub-leafset exhibit the same topology for Trp.

Lemma 1 (Tree consistency condition). *Let $\mathcal{G} = \{T_1, T_2, \cdots, T_n\}$ be a set of gene trees for a set of gene families organized into a set $\mathcal{X}$ of syntenies, and let S be the species tree. If a Super-Reconciliation $R(\mathcal{G}, S)$ exists, then the set of corresponding synteny trees $\{\tilde{T}_1, \tilde{T}_2, \cdots \tilde{T}_n\}$ is consistent.*

Proof. By definition, a Super-Reconciliation $R(\mathcal{G}, S)$ displays $\tilde{T}_i$, for all $1' \leq i \leq n$, as $R(\mathcal{G}, S)$ is an extension of each tree. Thus, for any triplet $Trp = \{X_1, X_2, X_3\}$ of $\mathcal{X}$, if $\tilde{T}_i$ and $\tilde{T}_j$ contain the triplet Trp as a sub-leafset, then $R(\mathcal{G}, S)$ displays both $\tilde{T}_i|_{Trp}$ and $\tilde{T}_j|_{Trp}$. In other words, $\tilde{T}_i|_{Trp}$ and $\tilde{T}_j|_{Trp}$ are label-isomorphic□

The consistency problem of rooted trees has been widely studied. The BUILD algorithm [2] can be used to test, in polynomial-time, whether a collection of rooted trees is consistent, and if so, construct a compatible, not necessarily fully resolved, supertree, i.e. a tree displaying them all. This algorithm has been generalized to output all compatible minimally resolved supertrees [8, 20, 23], which may be exponential in the number of genes.

The following theorem makes the link between a supertree and a reconciliation.

Theorem 1. *Let $\mathcal{G} = \{T_1, T_2 \cdots, T_n\}$ be a set of trees for a set of families organized in an order consistent set of syntenies $\mathcal{X}$, and S be the species tree. Let $\tilde{\mathcal{G}} = \{\tilde{T}_1, \tilde{T}_2 \cdots, \tilde{T}_n\}$ be the set of synteny trees corresponding to those in $\mathcal{G}$. If $\tilde{\mathcal{G}}$ is a consistent set of trees then:*

1. *A Super-Reconciliation $R(\mathcal{G}, S)$ is an extension of a supertree for $\tilde{\mathcal{G}}$;*
2. *Any supertree is the "backbone" of a Super-Reconciliation. Namely, for any supertree $\tilde{T}$ for $\tilde{\mathcal{G}}$, there is a Super-Reconciliation $R(\mathcal{G}, S)$ which is an extension of $\tilde{T}$.*

The first statement of Theorem 1 follows from Lemma 1. As for the second statement, we will prove it implicitly in Section 5 by providing an algorithm that yields a minimum cost reconciliation on any supertree.

Following Theorem 1, the problem reduces to finding a supertree for the set of synteny trees minimizing the number of segmental duplications and losses. A natural algorithm for the Super-Reconciliation problem follows:

1. Explore the space of all order consistent ancestral syntenies A for $\mathcal{X}$;
2. Explore the space of all supertrees $\tilde{T}$ for $\tilde{\mathcal{G}}$;
3. Find a Super-Reconciliation of minimum cost which is an extension of $\tilde{T}$ with A as an ancestral synteny;
4. Among all Super-Reconciliations and all ancestral syntenies, select the ones leading to the minimum cost.

Step 1 is discussed in Supplementary material and Step 2 has been discussed in this section. Before developing an algorithm for Step 3, which is the purpose of Section 5, we begin by analyzing the theoretical complexity of the Super-Reconciliation problem.

4 Complexity of the Super-Reconciliation Problem

We have recently considered the problem of finding a supertree of a set of gene trees minimizing the classical single gene duplication and single gene duplication and loss distances. The problem has been shown NP-hard for the duplication distance, and exponential-time algorithms have been developed for both distances. For segmental duplications only, the hardness of Super-Reconciliation is almost immediate from the results of [18]. For both duplications and losses, the problem remains NP-hard, although the proof is far more technical. Here we give the simpler proof of hardness for minimizing duplications only, and refer the reader to the Supplementary material for the NP-hardness proof for minimizing segmental duplications *and* losses.

Theorem 2. *The* Super-Reconciliation *problem is NP-hard for the duplication cost. Furthermore, the minimum number of duplications is hard to approximate within a factor $n^{1-\epsilon}$ for any $0 < \epsilon < 1$, where n is the number of syntenies in the input.*

Proof. The hardness follows from that of the MinDup-Supertree problem, defined as follows. Given a species tree S and a set of gene trees $T_1, \ldots, T_k$, possibly with overlapping leafsets, MinDup-Supertree asks for a supertree T that displays $T_1, \ldots, T_k$ such that the LCA-reconciliation with T and S yields

a minimum number d of duplications. It was shown in [18] that it is NP-hard to approximate d within a factor $n^{1-\epsilon}$ for any $0 < \epsilon < 1$, where here n is the number of genes in $\Gamma = \bigcup_{i=1}^{k} \mathcal{L}(T_i)$.

To reduce MinDup-Supertree to the Super-Reconciliation problem, it essentially suffices to exchange the roles of genes and syntenies. More precisely, given an instance of MinDup-Supertree consisting of a species tree S and gene trees $T_1, \ldots, T_k$, we compute an instance of Super-Reconciliation as follows. The species tree is the same as S, and for each gene $g \in \Gamma$, we have a synteny X_g with $s(X_g) = s(g)$. Moreover for each gene tree T_i, we create an identical gene tree T_i', but in which each gene $g \in \mathcal{L}(T_i)$ is replaced by a unique gene g_{T_i} that belongs to synteny X_g (and hence $s(g) = s(g_{T_i}) = s(X_g)$). Thus the synteny tree $\tilde{T}_i$ for T_i' is obtained by replacing each leaf g of T_i by X_g. In particular, there are n syntenies. The order of the genes on the syntenies is arbitrary (since we are not counting segmental losses).

It only remains to show the correspondence between the solutions for the two problem instances. Suppose that the MinDup-Supertree instance admits a supertree T with d duplications when reconciled. Let $\tilde{T}$ be the synteny tree obtained from T by replacing each gene $g \in \mathcal{L}(T)$ by X_g. Because $s(g) = s(X_g)$, both T and $\tilde{T}$ have the same duplications under the LCA reconciliation, which is d. Conversely, if our Super-Reconciliation instance admits a synteny tree $\tilde{T}$ with d duplications, replacing each leaf X_g by g yields a supertree for the MinDup-Supertree instance with d duplications. Because the value of the solutions are preserved and $n = |\Gamma|$ is the number of syntenies, this reduction is approximation preserving and the hardness result follows. □

We state our second hardness result formally here.

Theorem 3. *The* Super-Reconciliation *problem is NP-hard for the Dup, fLoss and pLoss cost.*

5 A Super-Reconciliation for a supertree

In this section, we are given a set $\mathcal{G} = \{T_1, T_2, \cdots, T_n\}$ of consistent gene trees for a set of families $\mathcal{F} = \{\Gamma_1, \Gamma_2, \cdots, \Gamma_n\}$ organized in an order consistent set of syntenies $\mathcal{X}$, and a species tree S for the set Σ of taxa containing the genes. In addition, we are given a supertree $\tilde{T}$ for the synteny trees $\tilde{\mathcal{G}} = \{\tilde{T}_1, \tilde{T}_2, \cdots, \tilde{T}_n\}$ corresponding to those in $\mathcal{G}$, and an order consistent ancestral synteny A for $\mathcal{X}$.

Given a Super-Reconciliation $R(\mathcal{G}, S)$ (R for short), because R is obtained from $\tilde{T}$ by grafting leaves, each node of $\tilde{T}$ is present in R. Hence we say that $x \in V(\tilde{T})$ has a *corresponding node* x' in R. More precisely, if $l \in \mathcal{L}(\tilde{T})$, then $l \in \mathcal{L}(R)$ also and the correspondence is immediate. If x is an internal node of $V(\tilde{T})$, the node x' of R corresponding to x is $lca_R(\{l : l \in \mathcal{L}(\tilde{T}[x])\})$. We show that, as in the traditional reconciliation setting, the nodes of R that are also in $\tilde{T}$ should be mapped to the lowest species possible. To simplify the argument, we will call an internal node a full loss if it is the parent of an $fLoss$ event.

Lemma 2. *Let $R(\mathcal{G}, S)$ be a Super-Reconciliation of minimum cost which is an extension of $\tilde{T}$. Let $x \in V(\tilde{T})$ and let x' be the node corresponding to x in $R(\mathcal{G}, S)$. Then $s(x') = lca_S(s(\mathcal{L}(\tilde{T}[x]))$.*

Proof. First observe that the statement is clearly true for the leaves. Assume that the statement is false. Now, let x be a node of $\tilde{T}$ such that its corresponding node x' does not satisfy the statement - moreover, choose x to be a minimal node with this property (meaning that for the children x_l and x_r of x, the corresponding nodes x'_l and x'_r in $R(\mathcal{G}, S)$ satisfy $s(x'_l) = lca_S(s(\mathcal{L}(\tilde{T}[x_l]))$ and $s(x'_r) = lca_S(s(\mathcal{L}(\tilde{T}[x_r])))$. Note that x must exist, since the statement is true for the leaves.

Now, we may assume that $s(x') \neq lca_S(s(x'_l), s(x'_r))$, as otherwise x' satisfies the lemma. Thus in S, there are at least k edges on the path from $s(x')$ to $lca_S(s(x'_l), s(x'_r))$, where here $k > 0$. It is not hard to verify that in this case, x' must be a duplication node, according to the definition of a reconciliation. This implies that there are at least k full losses on the path from x' to x'_l and at least k full losses on the path from x' to x'_r. Consider the Super-Reconciliation R' that is identical to $R(\mathcal{G}, S)$, with the exception that $s(x') = lca_S(s(x'_l), s(x'_r))$. Then the $2k$ losses on the paths between x' and x'_l and between x' and x'_r are not needed anymore, although if x' is not the root, k losses become necessary on the path between x' and y', where y' is the node corresponding to the parent y of x in $\tilde{T}$. Remapping x' cannot increase the number of duplications, and so we have saved k losses.

It remains to argue that the number of partial losses remains the same. But this is easy to see. We keep the same synteny assignment at nodes x', x'_l and x'_r (and y' if x' is not the root) as in $R(\mathcal{G}, S)$. If x' was a segmental duplication in $R(\mathcal{G}, S)$, we set x' to be a segmental duplication in R' as well. The number of partial losses on the paths between x' and x'_l, x'_r (and y') therefore remains the same as in $R(\mathcal{G}, S)$. □

We now show that speciation and duplication nodes are easy to identify. Essentially, we may set the events of internal nodes as in the classical LCA-mapping reconciliation. In what follows, assume that $\tilde{T}$ is reconciled under the LCA-mapping, and put $s(v) = lca_S(\mathcal{L}(s(\tilde{T}[v])))$ for every $v \in V(\tilde{T})$.

Lemma 3. *Let $R(\mathcal{G}, S)$ be a Super-Reconciliation of minimum cost which is an extension of $\tilde{T}$. Let $x \in V(\tilde{T})$ be an internal node of $\tilde{T}$ and let x' be its corresponding node in $R(\mathcal{G}, S)$. Moreover let x_l and x_r be the children of x. If $s(x_l)$ and $s(x_r)$ are separated in S, then x' is a speciation, and otherwise x' is a duplication.*

Proof. Let x'_l and x'_r be the nodes corresponding to x_l and x_r, respectively, in $R(\mathcal{G}, S)$. First, if $s(x_l)$ and $s(x_r)$ are not separated, then by Lemma 2, $s(x'_l)$ and $s(x'_r)$ are not separated, hence it is not possible for x' to be a speciation. Therefore x' must be a duplication.

Suppose instead that $s(x_l)$ and $s(x_r)$ are separated in S, but that x' is labeled by a duplication event $Dup(X, [i, j])$, where X is the synteny assigned at x'.

On the path from x' to x'_l, there may be some *pLoss* events and some nodes that were grafted owing to full losses. We may assume that all full loss events, if any, have occurred before the *pLoss* events on this path (i.e. nodes grafted from full losses are closer to x'). It is easily checked that this is without loss of generality, as this does not change the resulting synteny in x'_l. We shall make the same assumption with the path from x' to x'_r. Now, by Lemma 2, $s(x') = lca_S(s(x_l), s(x_r))$. Because x' is a duplication, the two children y_l, y_r of x' in $R(\mathcal{G}, S)$ must satisfy $s(y'_l) = s(y'_r) = s(x')$. Since $s(x'_l) \neq s(x') \neq s(x'_r)$, we have that $\{y_l, y_r\} \cap \{x'_l, x'_r\} = \emptyset$, and therefore y_l and y_r were grafted on $\tilde{T}$ due to full losses. If we label x' as a speciation $Spe(X, [1, |X|])$, these two full losses are not needed anymore, and by doing so we have one duplication less and two full losses less. Let Y_l and Y_r be the two syntenies that were assigned at y_l and y_r in $R(\mathcal{G}, S)$, respectively. Then $Y_l = X$ and $Y_r = X[i, j]$ or vice-versa (assume the former, without loss of generality). Suppose that y_r was an ancestor of x'_r in $R(\mathcal{G}, S)$, again without loss of generality. The substring $X[i, j]$ can be obtained from X by adding at most two partial losses on the path from x' to x'_r. The rest of the reconciliation can remain the same. To sum up, we have removed one duplication and two full losses, and inserted at most two partial losses to reproduce the effect of the segmental duplication. This contradicts the fact that $R(\mathcal{G}, S)$ is a reconciliation of minimum cost. □

From Lemma 3, it follows that we known event-type (Dup or Spe) for each internal node of the supertree $\tilde{T}$. It then remains to extend the tree with losses and infer the actual event at each node (i.e. the corresponding synteny and segment being duplicated or lost). It is easy to see that losses and segments affected by the events are fully determined by gene orders assigned to internal nodes. Therefore, the problem reduces to the classical "small phylogeny problem" most generally defined as follows: Given an alphabet Σ (nucleotides or amino-acids or genes), a distance on the set of words of Σ (edit distance for gene sequences or rearrangement distances for gene orders) and a tree T with leaves being words on Σ (extant gene sequences or gene orders), find the labeling of ancestral nodes (ancestral sequences or orders) minimizing the total cost of the tree. This cost is the sum of costs of each branch, which is the distance between the two words connected by the branch.

Here, we are given a synteny tree $\tilde{T}$ for a set $\mathcal{X}$ of syntenies on a set of gene families $\mathcal{F}$, and an ancestral synteny A which is an order of $\mathcal{F}$. We want to find a *synteny assignment* assigning a partial order on $\mathcal{F}$ to each node of $V(\tilde{T})$. We assume that the root r of $\tilde{T}$ is assigned the synteny A. It follows from Lemma 4 that, for two nodes u and v of $\tilde{T}$ with u being an ancestor of v, the synteny X_v assigned to v should be a subsequence of the string X_u assigned to u. A synteny assignment verifying this condition is called a *valid synteny assignment* for $\tilde{T}$.

For $v \in V(\tilde{T})$, define $d(v, X)$ as the minimum number of segmental duplications and losses induced by a synteny assignment on $\tilde{T}[v]$ with X being the assignment at v. The problem Small-Phylogeny for Syntenies Problem is to find an optimal assignment, i.e. an assignment leading to $d(\tilde{T}) =$

$\min_X d(r(\tilde{T}), X)$ for X belonging to the set of syntenies that are order consistent with $\mathcal{X}$.

Solving this problem can be done by dynamic programming by computing $d(v, X)$, for each $v \in V(\tilde{T})$ and each possible synteny X.

Let v be an internal node of $\tilde{T}$ and v_l, v_r be its two children. Let X, X_l, X_r be valid assignments for respectively v, v_l and v_r. Then X_l and X_r are subsequences of X. If v is a speciation, then all missing genes in X_l and X_r are the result of losses. Otherwise if v is a duplication, then for at most one of X_l and X_r, the missing prefix or suffix can be due to the partial duplication of a segment of X, and all other missing genes should be the result of losses. This motivates the definition of the following two variants of the loss distance between two syntenies.

Let X and Y be two syntenies with Y being a subsequence of X. We let $D^T(X, Y)$ denote the minimum number of segmental losses required to transform X to Y and $D^P(X, Y)$ the minimum number of segmental losses required to transform a substring of X to Y.

Theorem 4. *Let v be a node of $\tilde{T}$, X be a synteny and $\mathcal{S}(X)$ be the set of subsequences of X.*

- *If v is a leaf, then $d(v, X) = 0$ if X is the extant synteny corresponding to leaf v, and $+\infty$ otherwise;*
- *If v is a speciation with children v_l and v_r, then,*

$$\begin{aligned} d(v, X) = & min_{(X_l \in \mathcal{S}(X))}(D^T(X, X_l) + d(v_l, X_l)) + \\ & min_{(X_r \in \mathcal{S}(X))}(D^T(X, X_r) + d(v_r, X_r)); \end{aligned}$$

- *If v is a duplication node with children v_l and v_r, then*

$$d(v, X) = 1 + min \begin{cases} min_{(X_l \in \mathcal{S}(X))}(D^T(X, X_l) + d(v_l, X_l)) + \\ \quad min_{(X_r \in \mathcal{S}(X))}(D^T(X, X_r) + d(v_r, X_r)), \\ min_{(X_l \in \mathcal{S}(X))}(D^T(X, X_l) + d(v_l, X_l)) + \\ \quad min_{(X_r \in \mathcal{S}(X))}(D^P(X, X_r) + d(v_r, X_r)), \\ min_{(X_l \in \mathcal{S}(X))}(D^P(X, X_l) + d(v_l, X_l)) + \\ \quad min_{(X_r \in \mathcal{S}(X))}(D^T(X, X_r) + d(v_r, X_r)) \end{cases}$$

The above can be used to solve the Small-Phylogeny for Syntenies Problem by dynamic programming. To do this, one can simply traverse $\tilde{T}$ in post-order, and apply the recurrences of Theorem 4 at each node encountered. We finish this section by analyzing the complexity of this algorithm. Let $n = |V(\tilde{T})|$ and let t be the number of gene families involved in the Small-Phylogeny for Syntenies Problem instance. For a node $v \in V(\tilde{T})$ and a synteny X, there are $O(2^t)$ possible subsequences of X. The value of $d(v, X)$

thus depends on the $O(2^t)$ values for its left child v_l and the $O(2^t)$ values for its right child v_r. If these are known, then $d(v, X)$ can be computed in time $O(t2^t)$ (it is straightforward to check that D^T and D^P can be computed in time $O(t)$ - we omit these details).

Let us now consider the number of possible entries in our dynamic programming table. As shown in Lemma 4, the possible syntenies for X correspond to the subsequences of a topological sorting of an acyclic directed graph with t nodes. In the worst case, there are $O(2^t \cdot t!) = O(2^{t \log t + t})$ such syntenies. If follows that there are at most $O(n2^{t \log t + t})$ entries in the dynamic programming table, and each entry takes time $O(t2^t)$. It is known that if there are k possible topological sortings in a directed acyclic graph, then they can be enumerated in time $O(k)$ [22] (it is worth noting however that counting the number of such topological sortings in #P-complete [7]). Therefore, if t is not too large, then the above recurrences can solve the small phylogeny problem relatively quickly, even if n is large. Put differently, the Small-Phylogeny for Syntenies Problem is *fixed-parameter tractable* with respect to parameter t.

Corollary 1. *The* Small-Phylogeny for Syntenies Problem *can be solved in time* $O(t2^{t \log t + 2t}n)$.

6 Application

6.1 Simulated datasets

The dynamic programming algorithm has been implemented in C++ [4] and tested on balanced trees obtained from simulated evolutionary histories. Simulations have been performed according to five parameters: t, the number of gene families in the ancestral synteny; d, the maximum depth of the balanced tree; p_{dupl}, the probability for any given node to be a segmental duplication; p_{loss}, the probability for a loss to occur under any given node; p_{length}, where the probability for a loss to remove k genes is $P(X = k) = (1 - p_{length})^{k-1} p_{length}^{1-k}$, following a geometric distribution.

Simulations yield Super-Reconciliations leading to fully labelled trees. The input of the Super-Reconciliation algorithm is then obtained from those trees by removing loss nodes and synteny information on the internal, non-root nodes.

From an accuracy point of view (results not shown), as expected the larger the density of duplication and loss events, the further is the simulated history from a most parsimonious history, and thus from the inferred tree.

As for time-efficiency, values for inferring the Super-Reconciliation of a single tree, aggregated over 500 simulations per value of t, the size of the ancestral synteny (number of gene families), are given in Figure 2. Computations have been done on the "Cedar" cluster of Compute Canada with 32 *Intel 8160* CPUs operating at 2.10 GHz. As expected, running time exponentially increases with

[4] The program and simulations are available at:
https://github.com/UdeM-LBIT/SuperReconciliation

respect to parameter t. This prevented us from extending the simulations beyond an ancestral synteny of size 14, for which the Super-Reconciliation of a single tree of depth 5 required around 15 min. However, computational time follows a polynomial increase according to the size of the tree. As shown by the right diagram of Figure 2, for an ancestral synteny of size 5, simulations exhibit a running time of no more than few seconds for trees with depth up to 15, representing balanced trees with up to 2^{15} leaves.

With real biological datasets, it is more likely to have to deal with large gene families rather than large sets of gene families evolving in concert. Thus, the increase in running time according to the size of the ancestral synteny is not likely to be a strong limitation towards applying our Super-Reconciliation algorithm.

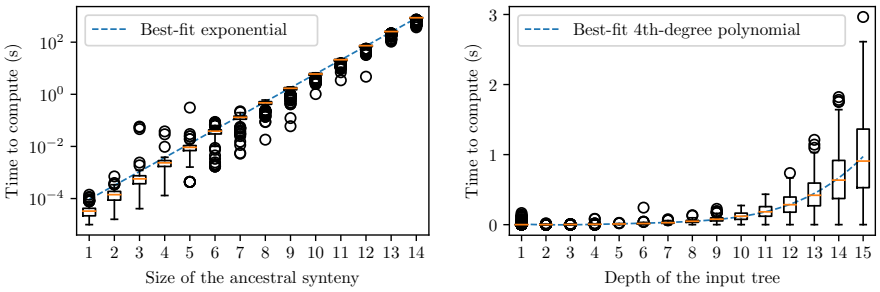

Fig. 2. Time-efficiency of the algorithm with respect to the size of the ancestral synteny (for $d = 5$) and the depth of the input tree (for $t = 5$), for $p_{dupl} = p_{loss} = p_{length} = 0.5$. Note that the leftmost graph uses a logarithmic scale.

6.2 The opioid system

The opioid receptors, important regulators of neurotransmission and reward mechanisms in mammals, offer an interesting proof of concept, as corresponding genes are present in clusters with conserved synteny in vertebrate genomes. Three genes for the opioid receptors (OPR) were identified and named OPRD1 (delta), OPRK1 (kappa) and OPRM1 (mu). A fourth gene was later identified (OPRL1) in rodents and human. In human, they are located on the four human chromosomes 1, 6, 8 and 20.

Previous studies have considered the duplication scenario explaining the evolution of the opioid receptor genes [10, 25, 26]. The main question was whether observed paralogons arose from the two whole genome duplication events, often called 1R and 2R, known to have occurred early in vertebrate evolution. By exploring regions surrounding the OPR genes in human, four syntenic regions, containing genes from three other families (NKAIN, SRC-B and STMN) apparently sharing a common history, were identified. From the analysis of individual gene trees (Neighbor-joining and quartet-puzzling maximum likelihood trees), conclusions associating the evolution of the opioid system related genes to the 1R and 2R events were drawn.

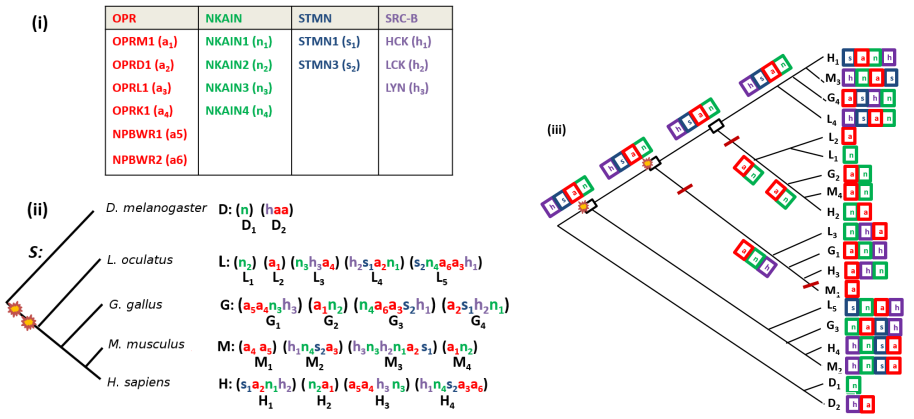

Fig. 3. (i) The four considered gene families. (ii) The considered species tree with the corresponding clusters: 19 in total involving 24 genes from the OPR family (genes named 'a'), 17 from the NKAIN family (named 'n'), 7 from the STMN family (named 's') and 13 from the SRC-B family (named 'h'). (iii) The Super-Reconciliation obtained form individual gene trees (not shown), and the induced duplication and loss history. Losses are indicated by red bars on the considered edges and duplications by rectangles. Yellow stars indicate the location of the 1R and 2R whole genome duplication events. Gene orders after removing duplicates (see text) are indicated on leaves, and chosen gene ordres for internal nodes are shown.

Here, we consider the same four gene families OPR, NKAIN, STMN and SRC-B, and further extend the OPR family with two neuropeptide NPBWR receptors, known to be closely related to the opioid receptors (Figure 3.(i)). Protein sequences and gene orders were downloaded from the Ensembl database (Release 92) [5] for the following five species: *Homo sapiens*, *Mus musculus*, *Gallus gallus*, *Lepisosteus oculatus* (spotted gar) and *Drosophila melanogaster*. Gene orders are given in Figure 3.(ii).

For each gene family, we built a multiple sequence alignments with ClustalW [16] (Gonnet weight matrix and gap opening and extension penalties respectively set to 10 and 0.2). Maximum likelihood gene trees were subsequently constructed for each family using MEGA7 [17] (Jones-Taylor-Thornton substitution matrix and uniform rates among sites). As some syntenies contained paralogs (multiple copies from the same gene family, for example synteny H_3 contains two 'a'), duplicates were removed in a way maximizing gene tree consistency. Although gene trees were still unconsistent, the overall clustering of gene copies was preserved among gene trees, and consistency could be attained after some local adjustments, using the species tree as reference.

The obtained Super-Reconciliation is given in Figure 3.(iii). Notice however that gene orders are far from being consistent. In fact, all considered genomes are separated by a considerable evolutionnary distance, and therefore, local rearrangements could have occured along each lineage-specific branch. Choosing

[5] https://useast.ensembl.org/index.html

the (h, s, a, n) order on every node of the tree and assuming rearrangements to occur at terminal edges, i.e. after duplication and loss events, leads to a history of three duplications and two losses before the speciation of bony fish and tetrapods, with two duplications correlating with the 1R and 2R tetraploidization events. This result is in agreement with previous studies on the evolution of the opioid receptor genes [10].

Further analysis, using more genes and species, is required to provide a more detailed scenario for the evolution of the opioid receptor genes. Our objective here however, was not to verify a given hypothesis, but rather to provide a proof of concept and explore the applicability and limitations of the proposed reconciliation model on real data.

7 Conclusion

We have presented a natural extension of the DL reconciliation model, which is the first effort towards the development of a unifying automated method for reconciling a set of gene trees. It leads to a variety of problems requiring to be analysed from a complexity and algorithmic point of view.

In contrast with the inference of tandem duplications, where gene orders is a key information as created gene copies should be adjacent to the original ones, order is not a central information for the Super-Reconciliation problem. In fact, as chromosomal segments resulting from transposed duplications can be placed anywhere in the genome, gene order in syntenies is not a required information for the reconstruction of the supertree. However, labeling the supertree in a way minimizing the number of segmental duplications and losses still requires the knowledge of an ancestral gene order.

If, as we have considered in this paper, rearrangements are forbidden, then a duplication and loss history does not always exist for a set of syntenies, as the corresponding gene orders may be inconsistent. One solution would be to minimally correct gene orders to ensure consistency, before applying the DL Super-Reconcilation model. Alternatively, an ancestral gene order can be inferred first, and all deviations from this order would be assumed to have occured at terminal edges. As it clearly appears from the opioid receptor genes example, rearrangements could hardly be ignored.

A future extension of this work will be to minimize the segmental duplication and loss events explaining the evolution of a set of syntenies evolving through speciation and segmental duplication, loss and rearrangements. In other words, we will infer gene orders leading to a most parsimonious history in terms of duplications and losses. The disagreement between the observed gene order at leaves and inferred orders can then simply be explained from rearrangements occuring after DL events. This is actually the approach we took to explain the supertree in Figure 3. Other natural extensions of this work would be to account for the possibility of paralogous genes inside synteny blocks and expand the reconciliation model to horizontal gene transfers.

References

1. A.A. Abbasi and K.H. Grzeschik. An insight into the phylogenetic history of hox linked gene families in vertebrates. *BMC Evolutionary Biology*, 7(239), 2007.
2. A.V. Aho, S. Yehoshua, T.G. Szymanski, and J.D. Ullman. Inferring a tree from lowest common ancestors with an application to the optimization of relational expressions. *SIAM J. Comput.*, 10(3):405- 421, 1981.
3. W. Ajmal, H. Khan, and A.A. Abbasi. Phylogenetic investigation of human fgfr-bearing paralogons favors piecemeal duplication theory of vertebrate genome evolution. *Molecular Phylogenetics and Evolution*, 81, 2014.
4. O. Akerborg, B. Sennblad, L. Arvestad, and J. Lagergren. Simultaneous bayesian gene tree reconstruction and reconciliation analysis. *Proceedings of the National Academy of Sciences USA*, 106(14):5714-5719, 2009.
5. M.S. Bansal, E.J. Alm, and M. Kellis. Efficient algorithms for the reconciliation problem with gene duplication, horizontal transfer and loss. *Bioinformatics*, 28(12):i283-i291, 2012.
6. S. Bérard, C. Gallien, B. Boussau, G.J. Szollosi, V. Daubin, and E. Tannier. Evolution of gene neighborhoods within reconciled phylogenies. *Bioinformatics*, 28(18):i382-i388, 2012.
7. Graham Brightwell and Peter Winkler. Counting linear extensions. *Order*, 8(3):225–242, 1991.
8. M. Constantinescu and D. Sankoff. An efficient algorithm for supertrees. *J. Classif.*, 12:101- 112, 1995.
9. J.P. Doyon, V. Ranwez, V. Daubin, and V. Berry. Models, algorithms and programs for phylogeny reconciliation. *Briefings in bioinformatics*, 12(5):392-400, 2011.
10. S. Dreborg, G. Sundstrom, T.A. Larsson, and D. Larhammar. Evolution of vertebrate opioid receptors. *Proceedings of the National Academy of Sciences USA*, 105(40):15487-15492, 2008.
11. D.E.K. Ferrier. Evolution of homeobox gene clusters in animals:the giga-cluster and primary vs. secondary clustering. *Frontiers in Ecology and Evolution*, 4(34), 2016.
12. J. Garcia-Fernàndez. The genesis and evolution of homeobox gene clusters. *Nature Reviews Genetics*, 6:881-892, 2005.
13. M. Goodman, J. Czelusniak, G.W. Moore, A.E. 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.
14. M. Hafeez, M. Shabbir, F. Altaf, and A.A. Abbasi. Phylogenomic analysis reveals ancient segmental duplications in the human genome. *Molecular Phylogenetics and Evolution*, 94, 2016.
15. Ian Holyer. The np-completeness of edge-coloring. *SIAM Journal on Computing*, 10(4):718–720, 1981.
16. J.D.Thompson, D.G. Higgins, and T.J. Gibson. Clustal w: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Research*, 22(22):4673-4680, 1994.
17. S. Kumar, G. Stecher, and K. Tamura. Molecular evolutionary genetics analysis version 7.0 for bigger datasets. *Molecular Biology and Evolution*, 33(7):1870-1874, 2016.

18. M. Lafond, A. Ouangraoua, and N. El-Mabrouk. Reconstructing a supergene-tree minimizing reconciliation. *BMC-Genomics*, 16:S4, 2015. Special issue of RECOMB-CG 2015.
19. T.A. Larsson, F. Olsson, G. Sundstrom, L.G. Lundin, S. Brenner, B. Venkatesh, and D. Larhammar. Early vertebrate chromosome duplications and the evolution of the neuropeptide y receptor gene regions. *BMC Evolutionary Biology*, 8(184), 2008.
20. M.P. Ng and N.C. Wormald. Reconstruction of rooted trees from subtrees. *Discrete Appl. Math*, 69:19- 31, 1996.
21. J. Paszek and P. Gorecki. Efficient algorithms for genomic duplication models. *IEEE/ACM Trans Comput Biol Bioinform.*, 2017.
22. Gara Pruesse and Frank Ruskey. Generating linear extensions fast. *SIAM Journal on Computing*, 23(2):373–386, 1994.
23. C. Semple. Reconstructing minimal rooted trees. *Discrete Appl. Math.*, 127(3), 2003.
24. J. Sjöstrand, A. Tofigh, V. Daubin, L. Arvestad, B. Sennblad, and J. Lagergren. A bayesian method for analyzing lateral gene transfer. *Sys.Biol.*, 63(3):409–420, 2014.
25. C.W. Stevens. The evolution of vertebrate opioid receptors. *Frontiers in bioscience: a journal and virtual library*, 14:1247-1269, 2009.
26. G. Sundstrom, S. Dreborg, and D. Larhammar. Concomitant duplications of opioid peptide and receptor genes before the origin of jawed vertebrates. *PLoS ONE*, 5(5), 2010.
27. G.J. Szöllősi, E., Tannier, V. Daubin, and B. Boussau. The inference of gene trees with species trees. *Syst.Biol.*, 64(1):e42–e62, 2014.
28. A. Tofigh, M. Hallett, and J. Lagergren. Simultaneous identification of duplications and lateral gene transfers. *IEEE/ACM Trans.Comput.BiolBioinform.*, 8(2):517-535, 2011.
29. W. Duchemin W, Y. Anselmetti, M. Patterson, Y. Ponty, S. Berard, C. Chauve, C. Scornavacca, V. Daubin, and E. Tannier E. DeCoSTAR: Reconstructing the ancestral organization of genes or genomes using reconciled phylogenies. *Genome Biol. Evol.*, 9(5):1312-1319, 2017.
30. L.X. Zhang. On Mirkin-Muchnik-Smith conjecture for comparing molecular phylogenies. *Journal of Computational Biology*, 4:177–188., 1997.
31. C. M. Zmasek and S. R. Eddy. A simple algorithm to infer gene duplication and speciiation events on a gene tree. *Bioinformatics*, 17:821– 828, 2001.

Supplementary material

7.1 Consistency of gene orders

Given a set of gene families $\mathcal{F} = \{\Gamma_1, \Gamma_2, \cdots, \Gamma_n\}$ organized into a set of syntenies $\mathcal{X}$, we define the *precedence graph* $\mathcal{P}$ as the directed graph with n vertices, each corresponding to a gene family of $\mathcal{F}$, such that a directed edge (i, j) between two vertices i and j exists iff there is a synteny $X = x_1 x_2 \cdots x_k$ of $\mathcal{X}$ containing a gene in Γ_i preceding a gene in Γ_j, i.e. there is a pair $1 \leq l_1 < l_2 \leq k$ such that $x_{l_1} \in \Gamma_i$ and $x_{l_2} \in \Gamma_j$.

If $\mathcal{P}$ is acyclic, then $\mathcal{P}$ is a Directed Acyclic Graph (DAG). In this case, there is a topological sorting for $\mathcal{P}$, i.e. a linear ordering X of vertices such that for every directed edge (i, j) in $\mathcal{P}$, i predeces j in X. Verifying if a directed graph is acyclic and finding a topological sorting of a DAG is a classical problem solvable in linear time.

The following lemma gives necessary and sufficient conditions for a set of syntenies to be order consistent and exhibits the set of possible ancestral syntenies.

Lemma 4 (Order consistency condition). *Let $\mathcal{F} = \{\Gamma_1, \Gamma_2, \cdots, \Gamma_n\}$ be a set of gene families organized into a set $\mathcal{X}$ of syntenies. Then $\mathcal{X}$ is order consistent iff the corresponding precedence graph $\mathcal{P}$ is acyclic. In this case, any topological sorting for $\mathcal{P}$ is an order consistent ancestral synteny for $\mathcal{X}$.*

Proof. The first part of the lemma follows from the fact that a directed graph has a topological sorting if and only if it is acyclic. The second part follows from the fact that, for any topological sorting A for $\mathcal{P}$ and any synteny X of $\mathcal{X}$, X is a subsequence of A, and thus X can be obtained from A through losses. □

The ancestral synteny A at the root of a Super-Reconciliation $R(\mathcal{G}, S)$ is an order on $\mathcal{F}$. Moreover, as the synteny at each internal node of $R(\mathcal{G}, S)$ is obtained from A through losses, a synteny at each internal node of $R(\mathcal{G}, S)$ should be a subsequence of A. More generally, for any two nodes x and y of $R(\mathcal{G}, S)$, where x is an ancestor of y, the synteny Y at y is a subsequence of the synteny X at x.

7.2 NP-hardness of the Super-Reconciliation problem

We reduce the problem of CUBIC 3-EDGE-COLORING to SUPER-RECONCILIATION. Given a graph $G = (V, E)$ in which each vertex has exactly 3 neighbors, the cubic 3-edge-coloring problem asks whether there exists a *proper coloring* of E with 3 colors, i.e. a partition of E into 3 sets $\{E_1, E_2, E_3\}$ such that for any vertex $v \in V$, the three edges incident to v all belong to a different E_i set. Note that if such a coloring exists, $|E_1| = |E_2| = |E_3| = |E|/3$. This problem was shown to be NP-hard in [15].

In what follows, for an integer k we will denote $[k] = \{1, 2, \ldots, k\}$. The father of a node v in a tree will be denoted $p(v)$ (p for 'parent'). Let $G =$

(V, E) be an instance of CUBIC 3-EDGE-COLORING, and denote $V = \{v_1, \ldots, v_n\}$. The ordering of the v_i vertices is not important, but must remain fixed for the duration of the proof. To describe our corresponding SUPER-RECONCILIATION instance, we first define the species tree S, which is illustrated in Figure 4. Let S' be a caterpillar on leafset $V \cup \{\alpha, \beta, \gamma\}$ (here a caterpillar is a binary rooted tree in which each internal node has at least one child that is a leaf), where the leaves appear in the order $(\alpha, v_1, v_2, \ldots, v_n, \beta, \gamma)$ when traversing from the deepest to the closest leaf to the root. The species α, β and γ are special species, and the $v_1, \ldots, v_n$ species are those corresponding to V. For each $i \in [n]$, denote $p_i := p(v_i)$, and $p_0 := \alpha, p_{n+1} := p(\beta)$. To obtain S, for every $i \in [n+1]$, graft a large number of new leaves, say n^{10}, on the branch $p_{i-1}p_i$. Thus there are now n^{10} new internal nodes on the path between p_{i-1} to p_i, and we denote this set of n^{10} internal nodes as W_i, and the set of n^{10} newly inserted leaves as W_i^{leaf} (Figure 4 only shows 5 of the W_i and W_i^{leaf} nodes, with W_2 and W_2^{leaf} shown explicitly). Note that S is a caterpillar with $(n+1)n^{10} + n + 3$ leaves.

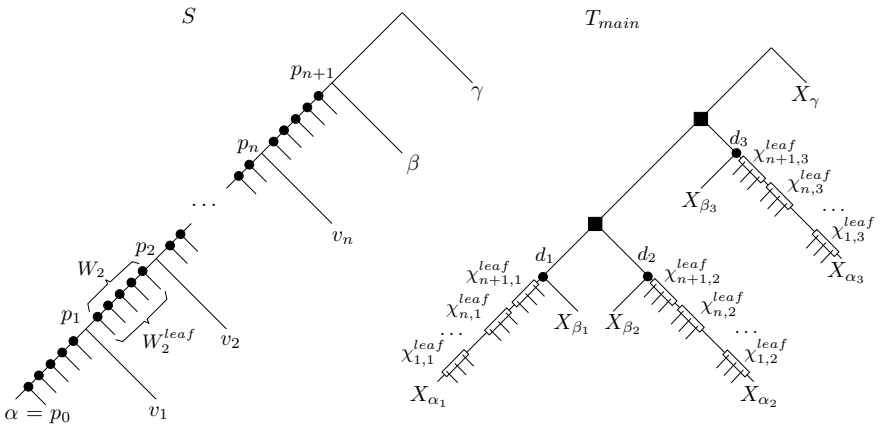

Fig. 4. The species tree S constructed in our reduction and the main synteny tree T_{main}. The $\mathcal{X}_{i,j}^{leaf}$ syntenies of T_{main} refer to the leaves, not the internal nodes. The black squares indicate duplications.

Now, denote $E' = \{(v_i, v_j), (v_j, v_i) : \{v_i, v_j\} \in E\}$, where we think of E' as the set of edges E, but where each edge appears in both directions. We define the set of syntenies

$$\mathcal{X} = \{X_\gamma\} \cup \{X_{\alpha_i}, X_{\beta_i} : 1 \leq i \leq 3\} \cup \mathcal{X}_{E'} \cup \bigcup_{i \in [n+1]} \bigcup_{j \in [3]} \mathcal{X}_{i,j}^{leaf}$$

where $\mathcal{X}_{E'} = \{X_{ij} : (v_i, v_j) \in E'\}$ and for each $i \in [n+1], j \in [3]$, $\mathcal{X}_{i,j}^{leaf}$ is a set of syntenies that has exactly one synteny for each member of W_i^{leaf}. We

put $s(X_{\alpha_i}) = \alpha, s(X_{\beta_i}) = \beta$ for each $1 \leq i \leq 3$, $s(X_\gamma) = \gamma$ and $s(X_{ij}) = v_i$ for each $(v_i, v_j) \in E'$. Hence the first subscript of a X_{ij} synteny indicates its species. Thus each species has 3 syntenies, with the exception of γ which has one. Observe that $|\mathcal{X}_{E'}| = 3n$.

We now define the set of input synteny trees (we do not bother with the actual genes, and so we give the synteny trees directly and omit the usual tilde symbol, e.g. we write T instead of $\tilde{T}$ with the understanding that T is a synteny tree whose corresponding gene tree has a unique gene at each of its leaves). Also, since each synteny tree corresponds to a gene family, we shall use the terms synteny tree and family interchangeably. There is one main synteny tree T_{main}, illustrated on Figure 4. To obtain it, for each $j \in [3]$, define a tree T^j_{main} as the unique synteny tree on leafset $\{X_{\alpha_j}, X_{\beta_j}\} \cup \bigcup_{i \in [n+1]} \mathcal{X}^{leaf}_{i,j}$ that has only speciations. Another way to view T^j_{main} is that it is obtained by taking a copy of S, by removing the $v_1, \ldots, v_n$ species and relabeling the species at the leaves by the syntenies listed in the preceding sentence. The tree T_{main} is then obtained by joining the T^j_{main} trees and X_γ as follows (using Newick notation):

$$T_{main} = (((T^1_{main}, T^2_{main}), T^3_{main}), X_\gamma)$$

Note that T_{main} has 2 duplication nodes. We will call d_1 and d_2 the two children of the lower duplication, and d_3 other duplication child (see Figure 4). The rough idea behind our reduction is that T_{main} has 3 subtrees that each contain a synteny from each species, except $v_1, \ldots, v_n$. These appear as losses in each T^j_{main} subtree. However, in χ we have three syntenies for each species $v_1, \ldots, v_n$, just enough to "fill-up" these losses. The main goal in our construction is to make this complete "fill-up" of the losses possible if and only if G is 3-colorable by making each T^j_{main} subtree represent a color. We will build additional input trees to prevent adjacent edges from filling losses in a common subtree.

But first, we also have another synteny tree T'_{main} defined as:

$$T'_{main} = (((X_{\alpha_1}, X_{\alpha_2}), X_{\alpha_3}), X_\gamma)$$

Note that T'_{main} does not provide any new information on the structure of a supertree, but will play a role in the gene ordering in the syntenies.

Then for each $v_i \in V$, let v_j, v_k, v_l be the neighbors of v_i and let

$$T_{ijk} = (((X_{ij}, X_{ji}), (X_{ik}, X_{ki})), X_\gamma)$$
$$T_{ijl} = (((X_{ij}, X_{ji}), (X_{il}, X_{li})), X_\gamma)$$
$$T_{ikl} = (((X_{ik}, X_{ki}), (X_{il}, X_{li})), X_\gamma)$$

For notational convenience, for all i, j, k we will say that $T_{ijk} = T_{ikj}$, i.e. both refer to the same tree. Note that for every edge $v_iv_j \in E$, X_{ij} occurs in exactly 4 trees (in 2 trees of the form T_{ijk} and 2 trees of the form T_{jik}). The set of synteny trees/families $\mathcal{G}$ contains T_{main}, T'_{main} and all the T_{ijk} trees.

It only remains to define the given gene orders for the extant syntenies. We will define these orders as strings on alphabet $\mathcal{G}$ directly instead of strings of genes. For $X \in \mathcal{X}$, let $t(X)$ be the set of synteny trees that X appears in. Observe that X_γ appears in every synteny tree, and thus there is only one possible ancestral sequence of genes. In other words, the X_γ synteny is a string Z over alphabet $\mathcal{G}$, and the order of every other synteny X is the subsequence of Z for the characters $t(X)$. We let Z be any string that begins with T_{main} and ends with T'_{main}. This completes the construction of our Super-Reconciliation instance.

Before we proceed, observe that if T is any supertree of $\mathcal{G}$, then T can be seen as a tree obtained by starting with T_{main}, then grafting some subtrees $T_1, \dots T_r$ successively onto some branches of T_{main}. To see this, let T' be the tree obtained from T by deleting every node that does not have a descendant in $\mathcal{L}(T_{main})$. Then T' is the same tree as T_{main}, but with some nodes of degree 2 denoted $t_1, \dots, t_r$ (excluding the root) that consist of the locations where the $T_1, \dots, T_r$ trees were grafted (note that the roots of T_{main} and T must be the same due to X_γ). For each $i \in [r]$, we will assume that the root of T_i was grafted onto T_{main} under t_i, i.e. $r(T_i)$ is a child of the node t_i. Note that $\{\mathcal{L}(T_1), \dots, \mathcal{L}(T_r)\}$ forms a partition of $\mathcal{X} \setminus \mathcal{L}(T_{main})$. Under the above view of T, we will think of a node x of T_{main} as also a node of T, whether x is internal or leaf.

We show that G admits a propoer 3-edge coloring if and only if our instance formed by $S, \mathcal{G}$ and order Z admits a Super-Reconciliation of total cost at most $3(n+1)n^{10} + 15n + 8$.

($\Rightarrow$) For the first direction, suppose that G admits a proper 3-edge coloring. Let E_1, E_2, E_3 be the underlying partition of the edges into 3 color classes. Let D_1, D_2 and D_3 be the subtrees of T_{main} rooted at d_1, d_2 and d_3, respectively. Notice that for each $l \in [3]$, the set $\{s(X) : X \in \mathcal{L}(D_l)\}$ is equal to $\mathcal{L}(S) \setminus V$. We "fill-up" each D_l subtree with a leaf from a synteny that belongs to v_j for every $v_j \in V$. More precisely, for each $l \in [3]$ and each $v_i \in V$, let v_iv_j be the edge of E_l that is incident to v_i. We graft the corresponding synteny X_{ij} (from species v_i) onto D_l on the branch that makes its parent a speciation (this location is unique). In this manner, every D_l subtree has exactly one synteny from every v_i species. Moreover, every synteny gets grafted onto T, no new duplications are created and T has no full losses. It is not difficult to verify that the resulting tree displays all input trees (the T_{ijk} trees are displayed because two X_{ij} and X_{ji} syntenies will be grafted under a D_l subtree different from X_{ik} and X_{ki}, since v_iv_j and v_iv_k belong to a different E_l set).

Our tree T has 2 duplications and no full losses. To count partial losses, we assign the string Z to every internal node. Let X be any leaf of T that belongs to a species in W_i^{leaf} for some $i \in \{1, \dots, n+1\}$. The X synteny only appears in T_{main} and T'_{main}, so because Z starts with T_{main} and ends with T'_{main}, we may add a single partial loss on the branch from X to its parent. This amounts to $3(n+1)n^{10}$ segmental losses. The syntenies X_{α_i} and X_{β_i} can be handled similarly with a single loss on the branch to their parent. As $i \in \{1, 2, 3\}$, this amounts to

6 losses. The X_γ leaf does not incur any losses. As for a synteny $X_{ij} \in \mathcal{X}_{E'}$, recall that X_{ij} appears in at most 4 gene families. This can be handled by using at most 5 segmental losses on the path between X_{ij} and its parent. As $|\mathcal{X}_{E'}| = 3n$, this adds at most $15n$ losses. In total, we the total cost is $3(n+1)n^{10} + 15n + 8$, which is conveniently the number that we predicted.

($\Leftarrow$) For the converse direction, let T be a supertree for $\mathcal{G}$ that yields a reconciliation of cost at most $3(n+1)n^{10} + 15n + 8$. We assume that each internal node is assigned a gene family sequence that is a subsequence of Z. We show how to obtain a proper edge coloring of G. The proof is divided into a series of Claims, the first one showing that every leaf in a W_i^{leaf} must incur a segmental loss.

Claim 1 *Let $X \in \mathcal{X}_{i,j}^{leaf}$ be a synteny for which $s(X) \in W_i^{leaf}$ for some $i \in [n+1]$ and $j \in [3]$. Moreover let p_X be the parent of X in T_{main}. Then in T, there is either a duplication or a partial loss on the path from p_X to X.*

Proof. Let Z_p be the gene family string assigned at p_X. Observe that in T, the p_X node has a descendant X_{α_i} for some $i \in [3]$. The family X_{α_i} appears in the trees T_{main} and T'_{main}, implying that both these families are in Z_p. Since X only appears in T_{main}, the T'_{main} character from Z_p must be lost on the path from p_X to X, either by a duplication or partial segmental loss. □

As a consequence of Claim 1, there are at least $3(n+1)n^{10}$ duplications and/or partial segmental losses in T. Our strategy is the following: we will show that if the X_{ij} syntenies are not setup to "fill up" every hole in the d_1, d_2 and d_3 subtrees of T as in our solution for the converse direction, then there must be at least n^{10} *full* losses in T (as opposed to partial losses). As these were not counted in Claim 1, this would imply that T has $3(n+1)n^{10} + n^{10} > 3(n+1)n^{10} + 15n + 8$ losses, a contradiction. Recall that we view T as a obtained from T_{main} by grafting subtrees $T_1, \ldots, T_r$ on nodes $t_1, \ldots, t_r$ that were inserted on branches of T_{main}. We next show that the trees that get grafted onto T_{main} to obtain T all consist of syntenies from a single species.

Claim 2 *Let $X_{ij} \in \mathcal{X}_{E'}$, and let T_h be the tree grafted onto T_{main} that contains X_{ij}. If T_h has another leaf $X_{kl} \in \mathcal{X}_{E'}$, then $k = i$.*

Proof. Suppose instead that $k \neq i$. Then $v_i = s(X_{ij}) \neq s(X_{kl}) = v_k$. We then have $s(r(T_h)) \geq lca_S(v_i, v_k)$. Assume without loss of generality that $k > i$. Observe that T_h cannot contain any leaf with a species in W_{i+1}^{leaf}, because all syntenies with a species in W_{i+1}^{leaf} are already in T_{main}. However, the path from $s(r(T_h))$ to v_i in S contains the set of nodes W_{i+1} (because $k > i$), where $|W_{i+1}| = n^{10}$. By the definition of reconciliation, each node in W_{i+1} must have at least one corresponding node in T_h on the path between $r(T_h)$ and X_{ij}, all of which must have a child that is a full loss in a node in W_{i+1}^{leaf}. It follows that T has at least n^{10} additional losses, a contradiction. □

Recall that T_{main} (and hence T) has two duplication nodes with children d_1, d_2 and d_3. These three nodes partition the leaves of T_{main} into 3 subsets. These will correspond to our edge colors. Towards this goal, for a synteny X_{ij}, we will say that X_{ij} is of color 1 (respectively color 2 and 3) if it is a descendant of d_1 (respectively of d_2 and d_3). Note that X_{ij} has at most one color, but may have none - which we prove to not be the case.

Claim 3 *For each synteny $X_{ij} \in \mathcal{X}_{E'}$, X_{ij} has a unique color.*

Proof. Suppose otherwise that there is some X_{ij} that has no color. Let T_h be the subtree grafted onto T_{main} that contains X_{ij}, with t_h the parent of $r(T_h)$. Since X_{ij} has no color, it follows that t_h must be an ancestor of d_1, d_2, d_3 or X_γ (all or some of these cases can hold simultaneously). In all cases, it is easy to see that $s(t_h) \geq lca_S(\alpha, \beta)$. Moreover by Claim 2, T_h has only leaves from the v_i species. It follows that on the path from t_h to X_{ij}, there is a loss for each node in W_{i+1}. Once again, this incurs n^{10} additional losses, a contradiction. □

We then show that syntenies from the same species get distinct colors, owing to the T_{ijk} trees.

Claim 4 *Let $X_{ij}, X_{ik} \in \mathcal{X}_{E'}$ be two distinct syntenies from the same species v_i. Then X_{ij} and X_{ik} do not have the same color.*

Proof. Supose that X_{ij} and X_{ik} have color 1, without loss of generality. Let $T_{i'}, T_{j'}$ and $T_{k'}$ be the subtrees grafted onto T_{main} that contain X_{ij}, X_{ji} and X_{ik}, respectively (where $t_{i'}, t_{j'}, t_{k'}$ are the parents of $r(T_{i'}), r(T_{j'}), r(T_{k'})$, respectively). By Claim 2, we know that $T_{i'} \neq T_{j'} \neq T_{k'}$, although $T_{i'} = T_{k'}$ is possible. Recall that we have the tree $T_{ijk} = (((X_{ij}, X_{ji}), (X_{ik}, X_{ki})), X_\gamma)$ in the input. Since T displays T_{ijk}, this implies that $t_{k'}$ cannot be a descendant of $lca_T(t_{i'}, t_{j'})$, and therefore $T_{i'} \neq T_{k'}$. Also, because X_{ik} is of color 1, $t_{k'}$ must be a descendant of d_1. Thus $t_{k'}$ is either (1) an ancestor of $lca_T(t_{i'}, t_{j'})$, or (2) $t_{k'}$ is on the path between a leaf $w \in \mathcal{X}_{l,1}^{leaf}$ and its parent $p(w)$ in T_{main}, where $l \in \{1, \ldots, n+1\}$. In case (1), the only way that T can display T_{ijk} is if X_{ki} belongs to $T_{k'}$, along with X_{ik}. This contradicts Claim 2. In case (2), let $T_{k''}$ be the subtree grafted on T_{main} that contains X_{ki}. Due to the T_{ijk} tree, $t_{k''}$ must also be on the path between w and $p(w)$. Thus in the subtree of T rooted at $lca_T(X_{ik}, X_{ki})$, there is at most one leaf other than X_{ik} and X_{ki} (namely w). This subtree must contain at least $n^{10} - 1$ losses, either for the W_{k+1} nodes if $i > k$, or the W_{i+1} nodes if $k > i$. We reach the same contradiction. □

It only remains to show that edge colors are consistent between their two directions.

Claim 5 *Let $X_{ij}, X_{ji} \in \mathcal{X}_{E'}$. Then X_{ij}, X_{ji} have the same color.*

Proof. Let v_k, v_l be the neighbors of v_i other than v_j. Suppose that X_{ij} and X_{ji} do not have the same color. If one of X_{ij} or X_{ji} is of color 3 and the other

of color 1 or 2, then because of the T_{ijk} tree in the input, X_{ik} and X_{ki} cannot be a descendant of any of d_1, d_2 or d_3. Hence they have no color, contradicting Claim 3. So we may assume that X_{ij} and X_{ji} are of color 1 and 2 (not necessarily respectively). Again because of the T_{ijk} tree, X_{ik} and X_{ki} must be of color 3. And because of the T_{ijl} tree, X_{il} and X_{li} must also be of color 3. But then, X_{ik} and X_{il} are both of color 3, contradicting Claim 4. □

We can now color the edges of E as follows: color v_iv_j with color $c \in \{1, 2, 3\}$ if and only if X_{ij} and X_{ji} have color c. By Claim 3 and Claim 5, each edge gets assigned a unique color. Two adjacent edges v_iv_j and v_iv_k get assigned the colors of X_{ij} and X_{ik}. By Claim 4, X_{ij} and X_{ik} have different colors. It follows that the edge coloring is proper, concluding the proof.