Gene Family Evolution by Duplication, Speciation and Loss

Cedric Chauve[1] Jean-Philippe Doyon[2] Nadia El-Mabrouk[3]

Keywords: gene families evolution, gene losses, reconciliation, algorithms.

[1] Department of Mathematics. Simon Fraser University. 8888 University Drive, V5A 1S6, Burnaby (BC), Canada. Tel: (1) 782-778-7091. Fax: (1) 778-782-4947. Email: cedric.chauve@sfu.ca

[2] Départment d'Informatique et Recherche Opérationnelle. Université de Montréal. CP 6128 succ. Centre-Ville, H3C 3J7, Montréal (QC), Canada. Tel: (1) 514-343-6111 (1865). Fax: (1) 514-343-5834. Email: doyonjea@iro.umontreal.ca

[3] Départment d'Informatique et Recherche Opérationnelle. Université de Montréal. CP 6128 succ. Centre-Ville, H3C 3J7, Montréal (QC), Canada. Tel: (1) 514-343-7481. Fax: (1) 514-343-5834. Email: mabrouk@iro.umontreal.ca

Abstract

We consider two algorithmical questions related to the evolution of gene families. First, given a gene tree for a gene family, can the evolutionary history of this family be explained with only speciation and duplication events? Such gene trees are called DS-trees. We show that this question can be answered in linear time, and that a DS-tree induces a single species tree. We then study a natural extension of this problem: what is the minimum number of gene losses involved in an evolutionary history leading to an observed gene tree or set of gene trees? Based on our characterization of DS-trees, we propose a heuristic for this problem, and evaluate it on a dataset of plants gene families and on simulated data.

1 Introduction

Background. Genes are the major building blocks of genomic sequences, containing the information necessary to produce all the proteins and non-coding RNAs of a cell. Genes, in a genome or across genomes, that are related by sequence or function similarity are called *homologs* and grouped into a *gene family*. The completed sequencing of a variety of genomes have revealed a high variability in the number of gene copies present in each genome. These changes in gene family sizes among species are due to successive phases of gene gains and losses (Demuth et al. 2006). In particular, gene duplication is a fundamental process in the evolution of species (Ohno 1970), and especially in eukaryotes (Lynch and Conery 2000, Eichler and Sankoff 2003, Dujon et al. 2004, Cotton and Page 2005, Blomme et al. 2006, Wapinski et al. 2007, Hahn et al. 2007), which is believed to play a leading role in the creation of novel gene functions. Several processes have been described to account for the origin of gene duplicates, ranging from the duplication of a single gene to the duplication of the entire genome (Durand and Hoberman 2006).

Understanding the evolution of gene families is a fundamental problem that has several applications (see (Durand et al. 2006) and references there). For example, it can help to distinguish between orthologs and paralogs: *orthologs* are copies that are directly related through speciation, while *paralogs* are copies that have evolved by duplication. This is an important question for functional annotation of genes, as it is believed that pairs of orthologs are more likely to have similar functions. Moreover, understanding the evolution of gene families can help establishing unambiguous one-to-one mappings between pairs of genomes, which is in general a hard computational problem (Blin et al. 2007) and a critical prerequisite for phylogenomics studies based on gene order and genomic rearrangements.

As the notion of orthology and paralogy is directly related to the history of speciation and duplication events during genomes evolution, a natural way to distinguish between the two types of gene homologs is to infer these events from the phylogenetic tree of a gene family, also called a gene tree. This question has been first considered in the case of a well established species tree. It can be described as "fitting a gene tree into a species tree", which is not obvious due to the possible incongruence between the two trees (Goodman et al. 1979,Guig o et al. 1996). This problem has been widely studied (Page 1994, Ma et al. 2000, Hallett and Lagergren 2000, Bonizzoni et al. 2005, Gorecki and Tiutyn 2006, Durand et al. 2006). The main algorithmic approach developed to solve it, the gene tree/species tree *reconciliation*, allows to identify the duplication and gene loss events with respect to the speciation events in the species tree. It is based on a mapping of the gene tree into the species tree that can be computed in linear time (Zhang 1997) (see also (Page 1998,Zmasek and Eddy 2001, Chen et al. 2000) for other implementations).

In the more general case where the species tree is unknown, a natural question is to infer a species tree from a gene tree, or more generally a set of gene trees, that optimizes a given criterion. Several criteria have been considered, either probabilistic (Arvestad et al. 2004) or combinatorial, like minimizing the number of duplications or the number of duplications plus losses. In particular, the problems of computing, for a given set of gene trees, the species tree inducing the minimum number of (1) duplication events and (2) duplication+loss events, have been shown to be NP-complete (Ma et al. 2000) but fixed-parameter tractable (Hallett and Lagergren 2000). Surprisingly, as far as we know, minimizing the number of gene losses (independently from the number of duplications) has never been considered. Though related to each other, the two combinatorial criteria (minimizing duplications versus minimizing losses)

are not equivalent: it may exist an evolutionary history leading to a given gene tree that minimizes the number of duplications but not the number of losses. The inverse proposition is also true, though an example of a history minimizing the number of losses but not the number of duplications is harder to find, suggesting that minimizing losses seems to be a more constraining criterion. The goal of this paper is to propose a first investigation of this combinatorial criterion for analyzing gene families.

Overview of our results. We start from a decision question, motivated among others by the importance of the duplication and speciation events to infer co-orthologs: given the gene tree of a specific gene family, can this gene tree be explained using only duplication and speciation events (e.g. without gene losses)? If a gene tree can be explained by a such a duplication/speciation history, we call it a DS-tree. The next question we address is the following: what is the minimum number of gene losses allowing to explain an observed gene tree that is not a DS-tree? The rationale for considering this question is that, given a gene tree, inferring the true gene loss events and their position in the tree gives a DS-tree that totally explains the evolutionary history of a gene family; as it is of course impossible to infer true gene loss events, we attack this problem following a parsimony approach. We show that this problem is equivalent to finding a species tree that minimizes the number of gene loss events in the combinatorial framework of reconciliation between a gene tree and a species tree, and then complements naturally the previous approaches based on duplications or duplications and losses (Ma et al. 2000, Hallett and Lagergren 2000, Chen et al. 2000). We propose a heuristic for this problem, that can be applied either to a single gene tree or to a set of gene trees that represent a set of gene families.

The natural application of our algorithms is to infer, from a set of gene trees, a species tree for the considered genomes, more precisely a species tree that minimizes the number of gene losses. However, from a practical point of view, considering individual gene families independently could be useful to detect such families that have evolved without or with few gene losses, and thus likely to be under very strong selection pressure. For example (Danchin et al. 2006) identified 11 gene families present in the whole eukaryotic spectrum (from yeasts to bilaterian animals), but not in mammals or vertebrates. This can be explained with a single loss event in the branch from the last common ancestor of mammals or vertebrates. It is important to notice that, when applied in such a way to single gene families, our approach is likely to generate false positive, i.e. gene families that can be explained with few gene losses but with a wrong species tree. However the gene families that have evolved with few gene losses will be detected as such by a method computing the minimum number of gene loss events, even if here too there is no guaranty on the correctness of the induced species tree and inferred gene losses. Altogether, our algorithms propose a first attempt to detect such gene families when no species tree is given.

Our results are presented as follows. In Section 3, we give two combinatorial characterizations of a DS-tree and a simple linear time recognition algorithm. We also show that a DS-tree induces a unique species tree compatible with an evolutionary history with no gene loss. In Section 4, we consider gene trees that are not DS-trees. Using the combinatorial framework developed in the study of DS-trees and the reconciliation approach, we propose a polynomial time and space algorithm that computes an upper bound on the number of gene loss events needed to explain a gene tree or a set of gene trees. The complexity of the problem of minimizing the number of gene losses is unknown, but it is related to Camin-Sokal parsimony which suggests it is a hard problem. Though the heuristic we propose does not provide a guaranteed

approximation ratio, its interest is twofold. First it is based on a combinatorial framework that complements the classical mapping approach, and provides insights on the problem of inferring gene losses and its complexity. Second, the computed number of gene losses can be used as an initial value that, in our experiments, greatly accelerated the running time of a branch-and-bound algorithm.

Finally, in Section 5, we apply our algorithms to two datasets: a dataset of plant gene families, computed from ESTs data, taken from (Sanderson and McMahon 2007), and a dataset of simulated gene trees based on a study of 12 *Drosophilia* species (Hahn et al. 2007). These two datasets are interesting for opposite reasons. The plants dataset contains gene families from seven angiosperms that are probably incomplete due to several reasons, among those the fact that they come from ESTs data. Moreover, the inferred gene trees are not guaranteed to be correct. However, a species tree is widely accepted. Our heuristic does not provide a realistic species tree, but the optimal species tree is very close to the accepted one. The simulated dataset provides a set of correct gene trees, that have evolved with relatively few gene losses. On this dataset, our heuristic performs very well, and the species tree that minimizes the number of gene losses is the correct species tree.

2 Definitions and problems statements

2.1 Definitions

Trees. Let $\mathcal{G} = \{1, 2, \cdots, g\}$ be a set of integers representing g different species (genomes). A *species tree* on $\mathcal{G}$ is a binary tree with exactly g labeled leaves, where each $i \in \mathcal{G}$ is the label of a single leaf. A *gene tree* T on $\mathcal{G}$ is a binary tree with labeled leaves, where each leaf is labeled by an integer from $\mathcal{G}$. It is a formal representation of a phylogenetic tree of a gene family, where each leaf labeled i represents a member of the gene family located on genome i. The size of a gene tree is its number of leaves. For a given vertex x of T, we denote by T_x the subtree of T rooted at x, and by $L(x)$ the subset of $\mathcal{G}$ defined by the labels of the leaves of T_x. $L(x)$ is called the *genome set of* x. We also denote by x_ℓ and x_r the two children of x, if x is not a leaf, and by x_p its parent if x is not the root of T. An *expanded leaf* of T is a vertex x of T such that $|L(x)| = 1$, and $|L(x_p)| > 1$ or x is the root of T.

Evolutionary history. The following is a formal definition of a Duplication/Loss/Speciation history (or simply a DLS-history) leading to a given gene tree.

Definition 1. A *DLS-history* of length n for a gene tree T on $\mathcal{G}$ is a sequence of gene trees $\mathcal{T} = (T^0, T^2, \cdots T^n)$ such that:

1. T^0 is a tree with a single vertex x labeled 1, and $T^n = T$;
2. For $0 < k < n$, one of the three following situations holds:
   (a) *Speciation event:* For given $i \in \mathcal{G}$ that label a leaf of T^k and $j \in \mathcal{G}$ that has never been used in a speciation event, T^{k+1} is obtained from T^k by adding two children y and z to each leaf x of T^k labeled i, removing the label of x and labeling one of the two new vertices by i and the other by j.
   (b) *Gene duplication event:* T^{k+1} is obtained from T^k by adding two children y and z to a leaf x, removing the label i of x, and labeling y and z with i.
   (c) *Gene loss event:* T^{k+1} is obtained from T^k by removing a leaf x and, is not the root of T^k, its parent y, and grafting the sibling of x to the parent of y if y.

If $\mathcal{T}$ is DLS-history for a gene tree T, we say that $\mathcal{T}$ *explains* T. Let $\mathcal{T}$ be a DLS-history explaining a gene tree T. Then, by construction, $\mathcal{T}$ leads to a unique species tree S induced by the speciation events. Note however that several DLS-histories can explain a same gene tree and induce different species trees. Figure 1 illustrates these concepts.

Obviously, an infinite number of DLS-histories can explain a single gene tree (due for example to silent gene duplications, i.e. gene duplication immediately followed by a gene loss). In the rest of this paper, we focus on particular DLS-histories: those minimizing the number of gene losses. Related to these DLS-histories, we introduce the notion of Duplication/Speciation histories.

Definition 2. A *DS-history* is a DLS-history with no gene loss event. A gene tree T is a *DS-tree* if it can be explained by a DS-history. A species tree induced by a DS-history for a gene tree T is said to be *DS-consistent with* T.

2.2 Problems statement

We now describe the two problems we address in our work. The first problem is the following:

DS-Tree Recognition Problem: Given a gene tree T on $\mathcal{G}$, is T a DS-tree?

We show in the next section that this problem can be solved in time and space linear in the size of T. Moreover, we show that given a DS-tree T, there is a single species tree that is DS-consistent with T. Although this property is not surprising, as far as we know it had never been proved formally.

The second problem we study in this paper is an optimization problem that we first state as follows: Given a gene tree T on $\mathcal{G}$, what is the minimum number of gene loss events in a DLS-history explaining T? To solve this problem, we relate it to DS-trees. Indeed, given a gene tree T that is not a DS-tree, it can be transformed into a DS-tree by a finite number of subtree insertions, where each insertion consists in grafting a new subtree onto an existing branch of T. It is immediate from the definition of DS-histories and DLS-histories that these subtree insertions correspond to gene loss events (see Fig. 2).

This leads to the following equivalent formulation of the optimization problem of inferring the minimum number of gene losses explaining a gene tree T:

Minimum Subtree Insertions Problem: Given a gene tree T, what is the minimum number of subtree insertions needed to transform T into a DS-tree?

Remark 1. The same problem can be considered for a set of gene families. Indeed, it is very likely that, given a set of gene families from a same set of genomes, optimizing the number of subtree insertions independently for each of these families can lead to inconsistencies as different species trees could be DS-consistent some of these DS-trees. This can happen for example with poorly defined gene families or widespread gene duplications. It would then make more sense to try to transform all these gene trees in DS-trees such that a single species tree is DS-consistent with all of them, by minimizing the total number of required subtree insertions. This can be reduced to consider the Minimum Subtree Insertions Problem with a single gene tree obtained by grafting all considered gene families trees under a root, possibly non-binary if there are more than two gene families. This approach only requires to extend the notion of DS-trees to gene trees with a non-binary root. We describe such an extension in Section 3 (Remark 2) and use it in our experiments (Section 5).

2.3 Reconciliation gene tree/species tree

We now show how the Minimum Subtree Insertions Problem can be stated in the classical gene tree/species tree reconciliation framework.

Suppose that a species tree S is already known for $\mathcal{G}$ and let T be a gene tree on $\mathcal{G}$. The gene tree/species tree reconciliation approach aims to infer an evolutionary history that has led to the gene tree T, based on a particular mapping (the LCA mapping) from the vertices of T to the vertices of S. This mapping maps every vertex x of T towards the Lowest Common Ancestor (LCA) of $L(x)$ in S. An internal vertex x is said to be a duplication vertex if $M(x_\ell) = M(x)$ and/or $M(x_r) = M(x)$. An internal vertex that is not a duplication is a speciation vertex. The definition of the number of gene losses is more intricate: (1) given two vertices y and y' of S such that y' is an ancestor of y, $d(y, y')$ is the number of vertices on the path from y to y', excluding y and y', (2) the number of losses associated to an internal vertex x of T, denoted ℓ_x, is

$$\begin{cases} 0 & \text{if } M(x) = M(x_\ell) = M(x_r) \\ d(M(x_\ell), M(x)) + 1 & \text{if } M(x_\ell) \neq M(x) \text{ and } M(x_r) = M(x) \\ d(M(x_r), M(x)) + 1 & \text{if } M(x_\ell) = M(x) \text{ and } M(x_r) \neq M(x) \\ d(M(x_r), M(x)) + d(M(x_\ell), M(x)) & \text{if } M(x_\ell) \neq M(x) \text{ and } M(x_r) \neq M(x) \end{cases} \quad (1)$$

and finally, the number of losses associated to the reconciliation of T with S, denoted by $\ell(T, S)$ is the sum over all internal vertices x of T of the number of losses ℓ_x associated to x. Note that the above definition of gene losses implicitly allows to locate them in a gene tree (see (Durand et al. 2006, Vernot et al. 2007) and Fig. 3).

In that reconciliation framework, we can state a natural optimization problem related to gene loss events: Given a gene tree T, find a species tree S such that $\ell(T, S)$ is minimum. The proposition below implies that this problem is in fact equivalent to the Minimum Subtree Insertions Problem.

Proposition 1. *Let T be a gene tree on $\mathcal{G}$ and S be a species tree on $\mathcal{G}$. Then $\ell(T, S) = k$ is the minimum number of subtree insertions needed to transform T into a DS-tree T' such that S is DS-consistent with T'.*

Proof. See Appendix. □

3 Recognizing a DS-tree

In this section, we propose two combinatorial characterizations of DS-trees. The first one follows a bottom-up (from the leaves to the root) approach, and is the basis of the linear-time recognition algorithm presented at the end of this section. The second characterization follows a top-down (from the root to the leaves) approach and is used in our heuristic for the problem of inferring the minimum number of gene losses required to recover a DS-tree from a given gene tree (Section 4). We first introduce a few notations and definitions.

Cherries: A *cherry* of T is a subset $\{i, j\}$ of $\mathcal{G}$ such that $L(x) = \{i, j\}$ for a given vertex x and the two children of x are expanded leaves ($L(x_\ell) = \{i\}$ and $L(x_r) = \{j\}$); x is said to be an *occurrence* of the cherry $\{i, j\}$, but when the context is clear, x is also said to be a cherry. An *expanded occurrence* of the cherry $\{i, j\}$ of T is a subtree T_x of T such that $L(x) = \{i, j\}$, $L(x_p) \neq \{i, j\}$ or x is the root of T, and every leaf of T_x belongs to a subtree rooted at an occurrence of the cherry $\{i, j\}$.

Definition 3. A cherry $\{i, j\}$ is said to be a *DS-valid cherry* for T if, for any expanded leaf x such that $L(x) = \{i\}$ or $L(x) = \{j\}$, x_p is an occurrence of $\{i, j\}$. In other words, no cherry of the form $\{i, k\}$ with $k \neq j$ or $\{j, k\}$ with $k \neq i$ exists in T.

If $\{i, j\}$ is a DS-valid cherry, we denote by $c(T, i, j)$ the gene tree on $\mathcal{G}\backslash\{i, j\} \cup \{g + 1\}$ obtained by replacing in T every expanded occurrence of the cherry $\{i, j\}$ by a single vertex (a leaf then) labeled $g + 1$.

Forests and borders: Let x be an internal vertex of T. The unordered pair $\{L(x_\ell), L(x_r)\}$ is called the *genomes partition associated to* x. We say that x *is valid* if and only if $L(x_\ell) \cap L(x_r) = \emptyset$. Let $\mathcal{F}$ be a forest, that is a set of one or more trees. We say that a set $\mathcal{X}$ of vertices of $\mathcal{F}$ is *covering* $\mathcal{F}$ if each leaf belonging to a tree of $\mathcal{F}$ is a descendant of a unique vertex of the set $\mathcal{X}$. We say that a vertex x is *higher* than a vertex z if z is a descendant of x. Let $\mathcal{B} = \{b_1, \ldots, b_k\}$ be the set of highest valid vertices of a forest $\mathcal{F}$: $\mathcal{B}$ is called a *border* if it is covering $\mathcal{F}$ and all the genomes partitions associated to the vertices of $\mathcal{B}$ are identical. Let $\mathcal{B}$ be a border of a forest $\mathcal{F}$, and $\{P_\ell, P_r\}$ be the partition of $\mathcal{G}$ induced by the children of the vertices of $\mathcal{B}$. We denote by $\mathcal{F}_\ell$ (resp. $\mathcal{F}_r$) the set of subtrees rooted in the children of vertices of $\mathcal{B}$ labeled by P_ℓ (resp. P_r) (see Figure 4 for an illustration).

Definition 4. A *DS-valid forest* is recursively defined as follows:

1. It is a set of expanded leaves or
2. It has a border and the forests $\mathcal{F}_\ell$ and $\mathcal{F}_r$ are DS-valid.

Theorem 1. *Let T be a gene tree on $\mathcal{G}$. The following statements are equivalent.*

1. *T is a DS-tree.*
2. *Either $|\mathcal{G}| = 1$, or for any cherry $\{i, j\}$, $\{i, j\}$ is a DS-valid cherry for T and $c(T, i, j)$ is a DS-tree on $\mathcal{G}\backslash\{i, j\} \cup \{g + 1\}$.*
3. *$\{T\}$ is a DS-valid forest.*

Proof. See Appendix. □

Corollary 1. *Let T be a DS-tree on $\mathcal{G}$. There exists a single species tree for $\mathcal{G}$ that is DS-consistent with T.*

Proof. See Appendix. □

Point 2 of Theorem 1 immediately translates into a simple algorithmic principle allowing to check whether a gene tree is a DS-tree. It is based on iteratively considering a cherry, checking its DS-validity, and then contracting all its occurrences into leaves and updating the species tree with the current cherry. We describe below a linear time and space algorithm based on this principle, taking as input a gene tree T on $\mathcal{G}$ with $|\mathcal{G}| = g$, and returning a species tree that is DS-consistent with T, if any.

```
ALGORITHM DS-RECOGNITION (T)
1.  LET m = g + 1 and S be a graph with of 2g − 1 vertices, labeled from 1 to 2g − 1, and no edge
2.  Perform a depth-first traversal of T, and let x be the current vertex
3.      IF x is a vertex with children x_ℓ and x_r with L(x_ℓ) = {i}, L(x_r) = {j} and i ≠ j THEN
4.          FOR EVERY vertex z_ℓ such that L(z_ℓ) = {i} DO
5.              LET z_r be the sibling of z_ℓ and z its parent
6.              IF L(z_r) = {j} THEN replace T_z by a leaf labeled m
7.              ELSE IF L(z_r) ≠ {i} THEN RETURN FALSE
8.          IF there remains a vertex x with L(x) = {j} THEN RETURN FALSE
9.          Connect in S the vertex labeled m with two children, the two vertices labeled i and j
10.         Increment m
11. RETURN S
```

Theorem 2. *Given a gene tree T with n vertices,* ALGORITHM DS-RECOGNITION *returns FALSE if and only if T is not a DS-tree, and the only species tree that is DS-consistent with T otherwise. It can be implemented to run in $O(n)$ time and space.*

Proof. See Appendix. □

Remark 2. As mentioned in Remark 1, it is a natural question to extend the results of this section to the case where the root of T is non-binary. This causes no problem for both characterizations of DS-trees. For the bottom-up characterization (point 2 of Theorem 1), this is taken into account by the case $|\mathcal{G}| = 1$, that does not consider the degree of the root. For the top-down characterization (point 3 of Theorem 1), it suffices to notice that the root of such a non-binary tree can not be part of the border of the forest $\mathcal{F} = \{T\}$, and then the forests $\mathcal{F}_\ell$ and $\mathcal{F}_r$ are well defined.

4 A heuristic for the Minimum Subtree Insertions Problem

We now describe an algorithm computing an upper bound on the minimum number of subtree insertions required to transform a gene tree T into a DS-tree.

Basically, due to the relationship between the Minimum Subtree Insertions Problem and the gene tree/species tree reconciliation problem (Proposition 1), for a gene tree T on a genome set $\mathcal{G}$, given any species tree S on $\mathcal{G}$, the LCA mapping induces a set of gene losses that explains T and then transforms T into a DS-tree. The problem is then to find such a species tree S that minimizes the number of gene losses. The heuristic we propose follows from the top-down characterization of DS-trees (point 3 of Theorem 1) and contains three steps:

- We first detect successive and disjoint sets of vertices in T, called levels, such that each level covers all leaves of T and is composed of either valid vertices (as defined in Section 3) and expanded leaves. Intuitively these levels contain vertices that indicate possible speciation events (for the valid ones) or unambiguous vertices (for the expanded leaves, whose history is then trivial and contains only duplications). For each level, we label each vertex x with a genome set $L'(x)$ that, roughly, contains $L(x)$ and describes the leaf set of x in a DS-tree T' obtained from T by subtree insertions and where x is a speciation vertex.
- We then compute from these levels, starting from the last one (the one that contains the last possible speciation events) to the first one (that contains the first possible speciation events), a species tree S that is compatible with all possible speciation events detected during the first phase. This is during this phase that we try, for each level, to minimize the number of gene loss events induced by the chosen species tree. This problem is related to the Camin-Sokal parsimony problem, and we describe it in Section 4.3.
- Finally, it follows from Proposition 1 that, given T and S, $\ell(T, S)$ gives an upper bound on the number of subtree insertions needed to transform T into a DS-tree.

Remark 3. The described algorithm differs from the algorithm published in a preliminary version of this work (Chauve et al. 2007) in two points. First we compute only a species tree and rely on the mapping defined in the reconciliation approach to infer gene losses events. Second, we propose a greedy heuristic to infer partial species trees (step 6 of Procedure Compute-Loss-Number, Section 4.2) instead of using arbitrary partial species trees.

4.1 The main algorithm

A set of vertices $\{x_1, \ldots, x_k\}$ is said to be *connected* if the intersection graph induced by the labels of these vertices (the vertices of the graph are the x_i's and two vertices are connected if their labels have a non-empty intersection) is connected. *Completing* the labels of a connected set $\{x_1, \ldots, x_k\}$ of vertices consists in adding to the label of every vertex x the subset $\cup_{i=1}^{k} L(x_i) \backslash L(x)$ of $\mathcal{G}$, leading to the new labeling $L'(x) = \cup_{i=1}^{k} L(x_i)$ (see Fig. 5).

We extend the notion of a valid vertex, defined in Section 3 for internal vertices, to expanded leaves: every expanded leaf is considered as a valid vertex. For a set of valid vertices $\mathcal{V}$, we call forest of $\mathcal{V}$, denoted $f(\mathcal{V})$, the set of subtrees of T rooted in the children of subset of the vertices of $\mathcal{V}$ that are not expanded leaves.

```
PROCEDURE RELABEL (T)
1.  LET F = {T} be the forest composed of the single tree T and k = 1;
2.  WHILE F is not a set of expanded leaves DO
3.      LET V_k be the set of highest valid vertices of F;
4.      Complete the labels of every maximal connected subset of V_k
5.      LET F = f(V_k);
6.      Increment k;
```

The successive sets of highest valid vertices of T considered in PROCEDURE RELABEL (the $\mathcal{V}_i$'s) are called the successive *levels of* T. Level $\mathcal{V}_1$ is the set of highest valid vertices, while, if T has k levels, level $\mathcal{V}_k$ is the last level. We denote by $r(T)$ the subset of $\mathcal{G}$ labeling the leaves of the forest $\mathcal{F}$ when $\mathcal{F}$ is a set of expanded leaves. An illustration of this procedure is given in Figure 5.

We now turn to the second step of our heuristic, that infers a species tree and the number f gene losses from the labeling L'. For a given level of T represented by a forest $\mathcal{F}$ of p trees $T_1, \ldots, T_p$ rooted at the vertices $x_1, \ldots, x_p$, we call the *partition of* $\mathcal{F}$ *by genome sets* the unique partition of $\mathcal{F}$ into subforests $\mathcal{F}_1, \cdots \mathcal{F}_p$ such that two trees T_i and T_j of $\mathcal{F}$ belong to the same subforest if and only if $L'(x_i) = L'(x_j)$ (x_i and x_j are respectively the roots of T_i and T_j). For $i \in \{1, \ldots, p\}$, we denote by $L'(\mathcal{F}_i)$ the subset of $\mathcal{G}$ that labels all roots of the trees belonging to the subforest $\mathcal{F}_i$. A species tree P for a genome set $\mathcal{G}$ *extends* a species tree P' for a subset $\mathcal{H}$ of $\mathcal{G}$ if P contains P' as a subtree.

```
PROCEDURE COMPUTE-LOSS-NUMBER (T, L')
1.  LET k be the number of levels of T and P_{k+1} = r(T);
2.  FOR i from k to 1 DO
3.      LET F be the forest composed of the tree rooted in vertices from V_i;
4.      LET F_1, ··· F_p be the partition of F by genome sets;
5.      FOR EVERY subforest F_j DO
6.          LET P_j be a species tree on L'(F_j), that extends every tree in P_{i+1}
            whose genome set is included in L'(F_j);
7.      LET P_i be the set of species trees P_1, ···, P_p.
8.  LET S be the only species tree in P_1.
9.  RETURN ℓ(T, S).
```

Theorem 3. PROCEDURE COMPUTE-LOSS-NUMBER *is well defined and computes an upper bound on the minimum number of subtree insertions needed to transform* T *into a DS-tree.*

Proof. See Appendix. □

We still did not indicate which species tree P_j should be chosen for the considered genome set $L'(\mathcal{F}_j)$. In a greedy approach, the natural choice would be to pick the species tree that extends the species trees present in $\mathcal{P}_{j+1}$ and induces the minimum number of gene losses when computing $\ell(T, S)$ (that correspond to subtree insertions located on the branches of nodes located between levels $\mathcal{V}_i$ and $\mathcal{V}_{i+1}$). In all generality this problem is as hard as our original problem, that deals with a single tree. The restriction that P_j should extend previously chosen species trees allows to reduce this problem to a conceptually simpler one, involving completing a set of small gene trees, that we attack with a greedy heuristic in Section 4.2.

Remark 4. It is important to point at the difference in nature between the two phases of our heuristic: the first one is not based on a combinatorial optimization principle, as it only detects

patterns in T that, following our characterization of DS-trees, indicate possible speciation events. Indeed, if, when processing a subforest $\mathcal{F}_j$ in steps 5-6 of Procedure Compute-Loss-Number the genome set $L'(\mathcal{F}_j)$ intersects exactly the genome sets of two species trees of $\mathcal{P}_{i+1}$, then there is a single choice for P_j. We rely on a greedy optimization phase when the pattern of speciations for a given $\mathcal{F}_j$ is ambiguous and there are more than one possible species tree that would be consistent with the possible speciations related to a level of valid vertices.

4.2 Completing a set of leaves and cherries

Problem definition: We now describe our approach to compute a species tree P_j in step 6 of Procedure Compute-Loss-Number. We first show that the problem is similar to the one of inferring a species tree that minimizes the number of gene losses in a set of gene trees, with the restriction that each of these gene trees is a cherry or a single leaf. This simpler modelization follows from Lemma 1 below.

Lemma 1. *Let T be a gene tree with k levels of valid vertices.*

1. *Every valid vertex in level $\mathcal{V}_k$ is either an expanded leaf or a cherry.*
2. *Let x be a child of a vertex belonging to level $\mathcal{V}_i$, $i < k$. Then all descendants of x belonging to level $\mathcal{V}_{i+1}$ have the same label L'.*

Proof. See Appendix. □

We now describe how we transform a level into leaves and cherries. Following Lemma 1, for a vertex x of $\mathcal{V}_i$ that is not an expanded leaf, we label its child x_ℓ (resp. x_r) with the genome set L' of its descendants belonging to level $\mathcal{V}_{i+1}$. If x is an expanded leaf, with $L(x) = \{i\}$, i also labels a leaf of a tree P of $\mathcal{P}_{i+1}$ and we define $L'(x)$ as $L'(x) = L(P)$. Then, if we denote by $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ the subset of $\mathcal{P}_{i+1}$ composed of the trees whose genome sets are included in $L'(\mathcal{F}_j)$, and by $\mathcal{X}$ the roots of the trees belonging to $\mathcal{F}_j$, every $x \in \mathcal{X}$ is one of the three following kinds of vertices.

- (Type 1) x is an expanded leaf with $L'(x) = L(Q_i)$ for some $Q_i \in \mathcal{P}$.
- (Type 2) $L'(x_\ell) = L'(x_r) = L(Q_i)$ for some $Q_i \in \mathcal{Q}$.
- (Type 3) $L'(x_\ell) = L(Q_i)$ and $L'(x_r) = L(Q_j)$, with $Q_i \neq Q_j$, $Q_i \in \mathcal{Q}$ and $Q_j \in \mathcal{Q}$.

Computing a species tree that extend all the species trees of $\mathcal{Q}$ and minimizes, among such trees, the number of subtree insertions is then equivalent to the same problem for the set of gene trees $\mathcal{T} = \{T_1, \ldots, T_p\}$ on $\{1, \ldots, m\}$ defined from $\mathcal{X}$ as follows: T_k is a leaf $\{i\}$ (a tree reduced to a single vertex labeled i) if the root x of T_k is a vertex of type 1 or 2 with $L'(x_k) = L(Q_i)$, or a cherry $\{i, j\}$ if x is a vertex of type 3 with $L'(x_\ell)$ and $L'(x_r)$ are equal to $L(Q_i)$ and $L(Q_j)$.

Given a species tree P on $\{1, \ldots m\}$, completing a leaf $\{i\}$ such that it becomes a DS-tree S on $\{1, \ldots m\}$ requires a number of subtree insertions $c(P, \{i\})$ equal to the number of branches on the path from i to the root of P, called the *depth* of i in P, denoted $\text{depth}_P(i)$. Completing a cherry $\{i, j\}$, if v is the lowest common ancestor of i and j in P, requires $c(P, \{i, j\}) = (\text{depth}_P(v) + \text{depth}_{P_v}(i) + \text{depth}_{P_v}(j) - 2)$ subtree insertions (branches on the paths from i to v, j to v and v to the root of P). (See Fig. 7). The cost of completing $\mathcal{T}$ given P is then equal to $c(P, \mathcal{T}) = \sum_{k=1,\ldots,p} c(P, L(T_k))$, and we are then interested in the following problem.

MINIMUM LEAF-CHERRY COMPLETION PROBLEM: Given $\mathcal{T}$, find a species tree P such that $c(P, \mathcal{T})$ is minimum.

Algorithm: We describe a heuristic solving the MINIMUM LEAF-CHERRY COMPLETION PROBLEM that is based on the sequential addition principle commonly used in heuristics to infer species trees for the parsimony criterion (Felsenstein 2004), but adapted to the particular structure of our problem. More precisely, we build a species tree in a greedy way, starting from a set of leaves, and iteratively joining a pair of subtrees, in such a way that at each step we have a lower bound on the final number of subtree insertions. We make a greedy choice for the pair of subtrees that are joined at each step: we chose the pairs that minimizes the increase of the lower bound.

Let $\mathcal{S} = \{S_1, \ldots, S_k\}$ be a set of leaf-disjoint trees on the genome set $\{1, \ldots, m\}$, such that the union of their leaves is equal to $\{1, \ldots, m\}$. We extend the definition of the cost of completing $\mathcal{T}$ as follows: $c(\mathcal{S}, \{i\})$ is the depth of i in the unique tree of $\mathcal{S}$ that has a leaf i, and $c(\mathcal{S}, \{i, j\}) = 0$ if no tree of $\mathcal{S}$ does contain both leaves i and j and $c(\mathcal{S}, \{i, j\}) = c(S_k, \{i, j\})$ if S_k is the tree of $\mathcal{S}$ containing both leaves i and j. It is then immediate that:

Lemma 2. *For every species tree P on $\{1, \ldots, m\}$ that extends all trees of $\mathcal{S}$, $c(P, \mathcal{T}) \geq c(\mathcal{S}, \mathcal{T})$, and $c(P, \mathcal{T}) = c(\mathcal{S}, \mathcal{T})$ if $\mathcal{S} = \{P\}$.*

To describe our heuristic, we need a last notation: if $\mathcal{S}$ is a forest with m trees, for every pair of distinct numbers i and j from $\{1, \ldots, m\}$, we denote by $\mathcal{S}_{i,j}$ the forest obtained from $\mathcal{S}$ by replacing S_i and S_j by a tree joining S_i and S_j under a root.

PROCEDURE COMPLETION-LEAF-CHERRY ($\mathcal{T}$)
1. LET $\mathcal{S} = \{1, \ldots, m\}$ the forest composed of m different leaves;
2. FOR k from 1 to $m - 1$ DO
3. Replace $\mathcal{S}$ by the forest $\mathcal{S}_{i,j}$ that minimizes $c(\mathcal{S}_{i,j}, \mathcal{T})$;
4. RETURN the only tree remaining in $\mathcal{S}$;

Proposition 2. *If $\mathcal{T}$ contains q trees on $\{1, \ldots, m\}$,* PROCEDURE COMPLETION-LEAF-CHERRY *($\mathcal{T}$) can be implemented in time $O(q + m^3)$ and space $O(m^2)$.*

Proof. See Appendix. □

Proposition 3. *The complete greedy heuristic, composed of* PROCEDURE RELABEL, PROCEDURE COMPUTE-LOSS-NUMBER *and* PROCEDURE COMPLETION-LEAF-CHERRY, *for a gene tree T of size n on a genome set of size g, can be implemented in time $O(g \times n + g^3)$ and space $O(g \times n)$.*

Proof. See Appendix. □

4.3 Link with Camin-Sokal Parsimony

We now describe how the MINIMUM LEAF-CHERRY COMPLETION PROBLEM is related to a phylogenetic reconstruction problem known as CAMIN-SOKAL PARSIMONY PROBLEM. In the CAMIN-SOKAL PARSIMONY PROBLEM (see (Felsenstein 2004) for example), given a set of binary characters, only species trees verifying the following property are considered: a character that is not present in a leaf is not present in its ancestors. In other words, characters

can only be gained. The goal is then to find a species tree that minimizes the total number of events, that is the number of character gains.

In the Minimum Leaf-Cherry Completion Problem, given a species tree P, each leaf (resp. cherry) present in $\mathcal{T}$ can be seen as character, weighted by the number of occurrences of this leaf (resp. cherry) in $\mathcal{T}$, that is present in all but one leaf (resp. two leaves) of P (for example a cherry $\{i, j\}$ will correspond to a character that is present in all m taxa but i and j). Then, for a given character (leaf or cherry) if the vertices of P are labeled 0 for the ones that are ancestor of a leaf not having this character and 1 otherwise. The number of subtree insertions for a given leaf or cherry is then the number of gains of the corresponding character in the labeled species tree P, and minimizing subtrees insertions is equivalent to minimizing characters gains as in the Camin-Sokal Parsimony Problem (see Fig. 8).

The Minimum Leaf-Cherry Completion Problem is then the restriction of the Camin-Sokal Parsimony Problem where each character is present in all but at most two taxa. Camin-Sokal Parsimony Problem has been shown to be NP-complete, but the hardness proof given in (Day et al. 1986) uses a reduction from Vertex Cover to Camin-Sokal Parsimony Problem instances where each character appears in exactly two taxa. Therefore, the NP-completeness of the Camin-Sokal Parsimony Problem does not imply the same result for the Minimum Leaf-Cherry Completion Problem, but, together with the hardness of other gene tree/species tree reconciliation problems (Ma et al. 2000), suggest that it is a hard problem.

5 Experimental results

The data and results of our experiments are available on a companion website:
`http//www.cecm.sfu.ca/~cchauve/JCB-CG07`.

5.1 A dataset of plant gene families

We describe the results obtained with the algorithms presented in the previous sections on the 577 gene families considered in (Sanderson and McMahon 2007), in a study of the phylogeny of seven angiosperm genomes from ESTs data[4]. Each of the 577 gene families was analyzed individually in a preliminary version of this work (Chauve et al. 2007). It was shown that most of them could be explained with very few gene losses (for example 333 gene families can be explained with no gene losses), but these results did not account for the fact that most families did span less than the 7 considered genomes. For example, 89 of the 333 DS-trees contain only 4 genes that span 3 genomes, while only 7 of the 59 gene families that span the 7 genomes are DS-trees.

In the present work, we are interested in the analysis of the 577 gene families together. We grouped the 577 gene trees under a single non-binary root and analyzed them both with our heuristic described in Section 4 and a branch-and-bound algorithm[5] that computed a species tree that minimizes the number of gene loss events event to explain the whole set of 577 gene families. The heuristic explained the 577 gene trees with a total of 4906 gene losses, compared to an optimal number of 3603 gene losses computed by the branch-and-bound algorithm. The species tree proposed by the heuristic is however very different from the accepted species tree, unlike the species tree proposed by the branch-and-bound that differs from the accepted species tree by two local rearrangements: *A. thaliana* and *S. tuberosum* are grouped into a single clade as are *O. sativa* and *Pinus* (see Fig. 9).

From these experiments, it appears first that minimizing the number of gene losses seems to be a good criterion to infer a species tree from a set of gene trees, even in the case of a possibly problematic dataset as can be gene trees obtained from widespread duplications histories and where gene families are obtained from ESTs data. Regarding the heuristic, it was useful from an algorithmical point of view as it computed an initial number of gene losses, that is less than two times the optimal and helped speed-up the branch-and-bound algorithm. Among the possible reasons for the poor result, in terms of species tree, obtained with the heuristic are either possible errors in computing gene families or gene trees, or the facts that the considered species are known for widespread duplications and that gene families could be incomplete due to the fact that they were computed from ESTs and not from sequenced genomes.

5.2 A simulated dataset

For our second experiment, we considered the phylogenetic tree described in Figure 1 of (Hahn et al. 2007), for 12 *Drosophilia* species, where three different rates of gene gain/loss are proposed, for different parts of the tree. We simulated 10 datasets of 100 gene families following a birth-and-death process as follows: starting from a single ancestral gene at the root of the species tree (with branch lengths) given in Fig. 1 of (Hahn et al. 2007), we did

[5] The algorithm we used is based on the sequential addition algorithm described in Section 4.2 and an improved version will be described in a future paper.

let this gene evolve along the branches of this species tree with a duplication rate of 0.02 (expected number of event by million years) and a gene loss rate of 0.02. We chose a gene gain/loss rate of 0.02 as it is slightly higher than the highest rate given in (Hahn et al. 2007) and because using the three rates given in (Hahn et al. 2007) resulted in datasets with too few gene losses to assess accurately our algorithms. The 1000 gene families contained a total of 3777 gene losses, distributed as described in Table 1.

For each of the 10 datasets, we proceeded as for the plant dataset and grouped the 100 gene families under a non-binary root. We then analyzed this tree with our heuristic and the branch-and-bound algorithm. In all cases the branch-and-bound inferred the correct species tree, with a number of gene losses lower than the (known) number of gene losses that happened during the evolution of the families, due, among others, to duplications immediately followed by a gene loss. In 6 of the 10 cases, the heuristic inferred the correct species tree and in the last 4 cases, the inferred species tree was exact up to a single local branch rearrangements (species trees are available on the companion website). The number of gene losses inferred by the heuristic and branch-and-bound by dataset are described in Table 2.

6 Conclusion

We showed in this paper that deciding if a gene tree can be explained without gene losses and lead to a unique species tree can be done efficiently. We also introduced the problem of inferring a species tree from a single gene tree or a set of gene trees by minimizing the number of gene loss events, that complements the classical gene tree/species tree reconciliation approaches that were based either on duplications or duplications and losses. Our preliminary experimental results, especially with simulated data where the exact answer is known, suggest that our approach can give good results, and should be compared with the two other reconciliation approaches based on minimizing the number of duplications and the number of duplications or gene losses.

Among the problems that our work suggest, the most natural is the complexity of the Minimum Subtree Insertions Problem, both for a single gene tree and for a set of gene trees. From the relationship between Camin-Sokal parsimony and the Minimum Leaf-Cherry Completion Problem, it is likely that it is NP-complete, but the problem is still open. Would it be NP-complete, it would then be interesting to ask if it is fixed-parameter tractable as the problems involving duplications (Hallett and Lagergren 2000) or if it can be approximated. Considering the problem of inferring gene loss events under a purely combinatorial point of view, and not from an optimization point of view, it would probably improve our heuristic to develop the approach used in the first step of the heuristic and based on detecting possible levels of duplications events. Handling gene trees that are not fully resolved is also a natural extension. Preliminary results on yeast gene families show that our approach needs to be generalized to non fully resolved gene trees, which would imply to consider non-binary species trees. Such problems have recently started to be investigated, both from algorithmical point of view (Chang and Eulensetin 2006, Berglund-Sonnhammer 2006, Vernot et al. 2007) and from a biological point of view (Fares et al. 2006, Hahn 2007).

Acknowledgments

This work was supported by grants from NSERC to C.C. and N.E.-M. and from SFU to C.C.

References

1. L. Arvestad, A.-C. Berglung, J. Lagergren and B. Sennblad. 2004. Gene tree reconstruction and orthology analysis based on an integrated model for duplications and sequence evolution, 326–335. In P.E. Bourne and D. Gusfield, eds, *Proceedings of the eighth Annual International Conference on Research in Computational Molecular Biology, RECOMB 2004*. ACM.
2. A.C. Berglund-Sonnhammer, P. Steffansson, M.J. Betts and D.A. Liberles. 2006. Optimal gene trees from sequences and species trees using a soft interpretation of parsimony. J. Mol. Evol., 63:240–250.
3. G. Blin, C. Chauve, G. Fertin, R. Rizzi and S. Vialette. 2007. Comparing genomes with duplications: a computational complexity point of view. *IEEE/ACM Trans. Comput. Biol. and Bioinformatics*, 4:523–534.
4. T. Blomme, K. Vandepoele, S. De Bodt, C. Silmillion, S. Maere, Y. van de Peer. 2006. The gain and loss of genes during 600 millions years of vertebrate evolution. *Genome Biol.* 7:R43.
5. P. Bonizzoni, G. Della Vedova and R. Dondi. 2005. Reconciling a gene tree to a species tree under the duplication cost model. *Theor. Comput. Sci.*, 347:36–53.
6. W.-C. Chang and O. Eulenstein. 2006. Reconciling gene trees with apparent polytomies, 235–244. In D.Z. Chen and D.T. Lee, eds*Computing and Combinatorics, 12th Annual International Conference, COCOON 2006*, vol 4112 of *Lecture Notes in Computer Science*. Springer.
7. C. Chauve, J.-P. Doyon and N. El-Mabrouk. 2007. Inferring a duplication, speciation and loss history from a gene tree (extended abstract), 45–57. In G. tesler and D. Durand, eds, *Comparative Genomics, RECOMB 2007 International Workshop, RECOMB-CG 2007*, vol. 4751 of *Lecture Notes in Computer Science*. Springer.
8. K. Chen, D. Durand and M. Farach-Colton. 2000. NOTUNG: a program for dating gene duplications and optimizing gene family trees. *J. Comput. Biol.*, 7:429–444.
9. J.A. Cotton and R.D.M. Page. 2005. Rates and patterns of gene duplication and loss in the human genome. *Proc. R. Soc. Lond. B*, 272:277–283.
10. E.G.J. Danchin, P. Gouret and P. Pontarotti. 2006. Eleven ancestral gene families lost in mammals and vertebrates while otherwise universally conserved in animals. *BMC. Evol. Biol.*, 6:5.
11. W.H.E. Day, D.S. Johnson and D. Sankoff. 1986. The computational complexity of inferring rooted phylogenies by parsimony. *Math. Biosci.*, 81:33–42.
12. J.P. Demuth, T. De Bie, J. Stajich, N. Cristianini and M.W. Hahn 2006. The evolution of mammalian gene families. *PLoS ONE*, 1:e85.
13. B. Dujon, D. Sherman, G. Fischer, *et al.* 2004. Genome evolution in yeasts. *Nature*, 430:35–44.
14. D. Durand, B.V. Haldórsson and B. Vernot. 2006. A hybrid micro-macroevolutionary approach to gene tree reconstruction. *J. Comput. Biol.*, 13:320–3354.
15. D. Durand and R. Hoberman. 2006. Diagnosing duplications-can it be done? *Trends Genet.*, 22:156–164.
16. E.E. Eichler and D. Sankoff. 2003. Structural dynamics of eukaryotic chromosome evolution. *Science*, 301:793–797.
17. O. Eulenstein, B. Mirkin and M. Vingron. 1997. Comparison of annotating duplication, tree mapping, and copying as methods to compare gene trees with species trees, 71–93. In B. Mirkin, F. McMorris, F. Roberts and A. Rzhetsky, eds, *Mathematical hierarchies and biology*, vol. 37 of *DIMACS Ser. Discrete Math. Theor. Comput. Sci.*. AMS.
18. M.A. Fares, K.P. Byrne and K.H. Wolfe. 2006. Rate asymmetry after genome duplication causes substantial long-branch attraction artifacts in the phylogeny of Saccharomyces species. *Mol. Biol. Evol.*, 23:245–253.
19. J. Felsenstein. 2004. *Inferring phylogenies.* Sinauer Associates.
20. P. Gorecki and J. Tiutyn. 2006. DLS-trees: a model of evolutionary scenarios. *Theor. Comput. Sci.*, 359:378–399.
21. M. Goodman, J. Czelusniak, G.W. Moore, A.E. Romero-Herrera and G. Matsuda. 1979. Fitting the gene lineage into its species lineage, a parsimony strategy illustrated by cladograms constructed from globin sequences. *Syst. Zool.* 28:132–163.
22. R. Guigó, I. Muchnik and T.F. Smith. 1996. Reconstruction of ancient phylogenies. *Mol. Phylogenet. Evol.* 6:189–213.
23. M.T. Hallett and J. Lagergren. 2000. New algorithms for the duplication-loss model, 138–146. In *Proceedings of the fourth Annual International Conference on Computational Molecular Biology, RECOMB 2000*. ACM.
24. M.W. Hahn. 2007. Bias in phylogenetic tree reconciliation methods: implications for vertebrate genome evolution. *Genome Biol.*, 8:R141.
25. M.W. Hahn, M.V. Han and S.-G. Han. 2007. Gene family evolution across 12 *Drosophilia* genomes. *PLoS Genet.*, 3:e197.

26. M. Lynch and J.S. Conery. 2000. The evolutionary fate and consequences of duplicate genes. *Science*, 290:1151–1155.
27. B. Ma, M. Li and L. Zhang. 2000. From gene trees to species trees. *SIAM J. Comput.*, 30:729–752.
28. S. Ohno. 1970. *Evolution by gene duplication.* Springer-Verlag.
29. R.D.M. Page. 1994. Maps between trees and cladistic analysis of historical associations among genes, organisms, and areas. *Syst. Biol.* 43:58–77.
30. R.D.M. Page. 1998. GeneTree: comparing gene and species phylogenies using reconciled trees. *Bioinformatics* 14:819–820.
31. M.J. Sanderson and M.M. McMahon. 2007. Inferring angiosperm phylogeny from EST data with widespread gene duplication. *BMC Evol. Biol.*, 7:S3.
32. B. Vernot, M. Stolzer, A. Goldman and D. Durand. 2007. Reconciliation with non-binary species trees, 441–452. In P. Markstein and Y. Xu, eds, *Computational Systems Bioinformatics. Proceedings of the Conference CSB 2007.* Imperial College Press.
33. I. Wapinski, A. Pfeffer, N. Friedman and A. Regev. 2007. Natural history and evolutionary principles of gene duplication in fungi. *Nature*, 449:54–61.
34. L.X. Zhang. 1997. On Mirkin-Muchnik-Smith conjecture for comparing molecular phylogenies. *J. Comput. Biol.*, 4:177–188.
35. C.M. Zmasek and S.R. Eddy. 2001. A simple algorithm to infer gene duplication and speciation events on a gene tree. *Bioinformatics*, 17:821–828.

7 Appendix.

Proof of Proposition 1

It follows from the definition of subtree insertions and gene losses in the reconciliation theory that at most k subtree insertions are needed to transform T into a DS-tree T' such that S is consistent with T' (see for example (Durand et al. 2006)). We then just need to show that it is impossible to transform T into a DS-tree that is DS-consistent with S with less than k subtree insertions.

Let T'' be a DS-tree obtained from T by less than k subtree insertions such that S is DS-consistent with T''. Moreover, we assume that among such DS-trees T'' is obtained form T using the minimum number of subtree insertions. Without loss of generality, we also assume that the k subtree insertions in T' are the ones given by the LCA mapping between T and S.

Let x be an internal vertex of T (x is then also a vertex of T' and T''), and y and z its children in T (also vertices in T' and T'' but it is possible that they are not children of x in these two trees due to subtree insertions), such that $T'_y = T''_y$, $T'_z = T''_z$ but $T'_x \neq T''_x$. Such a vertex obviously exists as T'' is different from T', but S is DS-consistent with both DS-trees. Let $u = M(y)$, $v = M(z)$ and $r = M(x)$, be the vertices associated to y, z and x by the LCA mapping M. Then, by definition of the LCA mapping M, S_u (resp. S_v, S_r) is DS-consistent with $T'_y = T''_y$ (resp. $T'_z = T''_z$, T'_x) that is a DS-tree. Moreover, by definition of DS-trees, there is a vertex w of S such that S_w is DS-consistent with T''_x. We now show that w is different from r and is an ancestor of r. If $w = r$, $T'_y = T''_y$ and $T'_z = T''_z$, together with the definition of the LCA mapping, imply that the subtree insertions induced by the LCA mapping are needed. The minimality of the number of subtree insertions to transform T into T", implies then that $T'_x = T''_x$, which contradicts our assumption. If r is an ancestor of w, S is not DS-consistent with T'' as w is not an ancestor of either u or v. Finally, if neither r is an ancestor of w nor w is an ancestor of r, both u and v are not descendants of w and S is not DS-consistent with T''. Hence w is an ancestor of r.

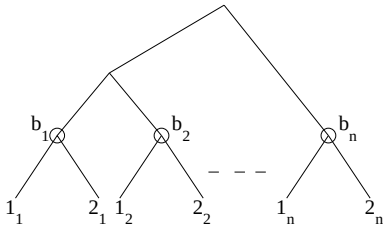
This implies that more subtrees have been inserted on the branches (x, y) and (x, z) of T to obtain T'' than to obtain T'. More precisely, some subtree of S has been inserted both on the branches (x, y) and (x, z). Such insertions could be avoided to transform T into a DS-tree, by being moved on the branch from x to its parent. It would then be possible to obtain a DS-tree such that S is DS-consistent with it in less subtree insertions than required to obtain T'', which contradicts the minimality of T''.

Proof of Theorem 1

'$1 \Rightarrow 2$' The results holds obviously for $g = |\mathcal{G}| = 1$. Assume $g > 1$, T is a DS-tree and let $T^1, \ldots, T^n$ be a DS-history explaining T, its last speciation event being from T^k to T^{k+1}, producing leaves labeled i and j, with $i < j$ (so $j = g_k + 1$). Without loss of generality, we can assume that all duplication events involving leaves with a label different from i and j occurred before this speciation event. It follows immediately from the fact that all later events are duplication events involving i or j that $\{i, j\}$ is a DS-valid valid cherry. Moreover up to relabeling leaves labeled $g + 1$ by i $c(T, i, j) = T^k$. So $c(T, i, j)$ can be explained by the DS-history $T^1, \ldots, T^k$, and then $c(T, i, j)$ is a DS-tree.
'$1 \Leftarrow 2$' Here again the base case ($g = 1$) is obvious. In the general case ($g > 1$), let $\{i, j\}$ be a DS-valid cherry, with $i < j$. By hypothesis, $c(T, i, j)$ is a DS-tree, so is the same tree where leaves labeled with $g + 1$ are replaced by i. Hence, $c(T, i, j)$ can be explained by a DS-history

$T^1, \ldots, T^k$. Next, to go from $c(T, i, j)$ to T, as $\{i, j\}$ is a DS-valid cherry, we replace each leaf i by a subtree where one child x_ℓ of the root x is such that $L(x_\ell) = \{i\}$, and the other child such that $L(x_r) = \{i\}$. This corresponds immediately to single speciation event producing leaves i and j followed by possible duplication events of leaves i and j, and then a DS-history explaining T.

'$1 \Rightarrow 3$': Let T be a DS-tree, and $\mathcal{T}$ be a duplication/speciation history for T. Let n be the number of gene copies existing in the ancestral genome preceding the first speciation event that gave rise to two genomes labeled 1 and 2, and let $\{1_1, 1_2, \cdots, 1_n\}$ (resp. $\{2_1, 2_2, \cdots, 2_n\}$) be the gene family in genome 1 (resp. genome 2). The gene tree resulting from the first speciation event has the following general shape:



As gene losses are not allowed in any DS-history leading to the DS-tree, each gene 1_i (resp. 2_i), for $1 \leq i \leq n$, is the ancestral copy of at least one gene in each genome resulting from subsequent speciation events of genome 1 (resp. genome 2). Therefore, all b_i left (resp. right) genome sets are identical, for $1 \leq i \leq n$, and correspond to the set of genomes descendant from genome 1 (resp. genome 2). Therefore the set $\{b_1, \cdots b_n\}$ corresponds to a border of T. A similar argument allows to prove that all subsequent forests have a border, which concludes this first part of the proof.

'$1 \Leftarrow 3$': Let T be a gene tree such that T has a border B of size n and its resulting forests $\mathcal{F}_\ell$ and $\mathcal{F}_r$ are DS-valid with borders $\mathcal{B}_\ell$ and $\mathcal{B}_r$ respectively of size n_ℓ and n_r. If T is a gene tree with $|\mathcal{G}| = 1$ and n leaves, then T is a DS-tree explained by a DS-history constituted by a sequence of $n-1$ gene duplications. Otherwise, by induction on $\mathcal{G}$, it is immediate to prove that the procedure DS-Construct below computes a DS-history for T.

PROCEDURE DS-CONSTRUCT (T, B, n)
1. Perform $n-1$ duplication events
2. Perform a speciation event
3. DS-Construct($\mathcal{F}_\ell$, $\mathcal{B}_\ell$, n_ℓ)
4. DS-Construct ($\mathcal{F}_r$, $\mathcal{B}_r$, n_r)

Proof of Corollary 1

We prove the statement by induction on $g = |\mathcal{G}|$, and using the characterization of DS-trees as given in point 2 of Theorem 1. If $g = 1$, the statement obviously holds. Let $g > 1$ and $\{i, j\}$ be a cherry for T. As T is a DS-tree, let $\{i, j\}$ be a DS-valid cherry of T. From the definition of a cherry, we can say that in every DS-history explaining T, there is a speciation event that produces i and j, followed by possible duplications of i and j. It follows that every species tree DS-consistent with T contains a vertex with two children that are leaves and labeled respectively with i and j. Finally, as $c(T, i, j)$ is a DS-tree on $g-1$ species, by induction there is a unique species tree DS-tree-consistent with $c(T, i, j)$, which implies that there is a unique species tree S that is DS-consistent with T.

Proof of Theorem 2

The fact that the algorithm checks whether T is a DS-tree follows immediately from Theorem 1 and the fact that steps 4 to 8 obviously check if the current cherry $\{i, j\}$ is DS-valid. To assess the time complexity, one can first notice that every vertex is visited a constant number of times, during the traversal of T (step 2) and in step 4, step 5 or step 8.

To get a linear time complexity we just need to ensure that steps 4 and 8 are performed in time linear in the number of visited vertices. To achieve this bound, we maintain during the algorithm a mapping V from $\{1, \ldots, 2g-1\}$ to subsets of vertices of T, such that, at any time in running the algorithm, $V(i)$ is the subset of vertices of T that root subtrees whose all leaves are labeled by i. More precisely, $V(i)$ is a linked list of all the vertices z of T such that $L(z) = \{i\}$. Moreover, we maintain a mapping W from each leaf of T to the cell of the lists of V that contain this leaf. V and W can be initialized during a single depth-first search in T. Then with V known, and provided the list $V(i)$ can be accessed in constant time (which is easy if V is an array of linked lists), the steps 4 and 8 can be performed in time linear in the number of visited vertices. Finally, if after step 6, z_r and z_ℓ are removed and z is replaced by a leaf labeled m, it suffices to remove z_r and z_ℓ from the lists $V(i)$ and $V(j)$ that contain them (which is easy using W), and z is added to the list $V(m)$ and $W(z)$ is set to the corresponding cell in this list, which can both be done in constant time. The linear space complexity is obvious.

Proof of Theorem 3

To prove that Procedure Compute-Loss-Number is well defined, we only need to prove that (1) in step 6 there exists a species tree P_j on the genome set $L'(\mathcal{F}_j)$ that extends every tree in $\mathcal{P}_{i+1}$ whose genome set is included in $L'(\mathcal{F}_j)$, and (2) $\mathcal{P}_1$ contains a unique species tree that is a species tree on $\mathcal{G}$.
(1) The statement obviously holds when $i = k$ as $\mathcal{P}_{k+1} = r(T)$ and then every element of $r(T)$ is a single vertex by construction. If $i \neq k$, as we defined one species tree in $\mathcal{P}_{i+1}$ for each part of the partition of $\mathcal{F}_i$ by genome sets, the genome sets of the trees in $\mathcal{P}_{i+1}$ are pairwise disjoint, which immediately implies the stated property.
(2) We first show that the partition by genome sets of $\mathcal{F}_1$ has only one part. Let x be a vertex of T such that x is the lowest common ancestor of two vertices y and z of $\mathcal{V}_1$ with $L'(y) \neq L'(z)$, and no descendant of x does satisfy the same property. If x is the parent of both y and z, then x is valid and should belong to $\mathcal{V}_1$, which contradicts the fact that it is ancestor of two vertices of $\mathcal{V}_1$. Otherwise (x is not the parent of both y and z), as all leaves of T are covered by vertices of $\mathcal{V}_1$, there are other descendants of x that belong to $\mathcal{V}_1$. All vertices of $\mathcal{V}_1$ that belong to the same subtree of T_x than y (resp. z) have the same label L' than y (resp. z), as otherwise an ancestor of y (resp. z) would belong to $\mathcal{V}_1$. Here again this contradicts the fact that x is not a valid vertex. So we can not have $L'(y) \neq L'(z)$ for any pair x and y of vertices of $\mathcal{V}_1$.
Finally, as the partition by genome sets of $\mathcal{F}_1$ has only one part, $\mathcal{P}_1$ contains a unique species tree S. By definition, the vertices of $\mathcal{V}_1$ covers all leaves of T and then S is a species tree on $\mathcal{G}$.

The fact that $\ell(T, S)$ is an upper bound for the minimum number of subtree insertions needed to transform T into a DS-tree follows from Proposition 1.

Proof of Lemma 1

(1) Otherwise, given a valid vertex that roots a subtree with at least three leaves with different labels, one of its descendant is itself a valid vertex and then T has at least one more level of valid vertices.
(2) We use induction on the number of descendants of x that belong to $\mathcal{V}_{i+1}$. If there is only one, the statement obviously holds. Otherwise, if there are two such vertices that are sibling, say x_ℓ and x_r, as their parent is not a valid vertex, $L(x_\ell) \cap L(x_r) \neq \emptyset$, and then, following Procedure Relabel $L'(x_\ell) = L'(x_r)$. By induction, we can then conclude. Finally, the fact that two such sibling vertices of level $\mathcal{V}_{i+1}$ exist follows from the fact that the descendants of x that belong to level $\mathcal{V}_{i+1}$ cover all leaves of T_x.

Proof of Proposition 2

We use an auxiliary weighted complete graph $\mathcal{L}$ and maintain the following invariants: (1) every vertex of $\mathcal{L}$ is a tree S_k of $\mathcal{S}$, (2) the weight of a vertex S_k, denoted by $w(k)$, is equal to the number of trees of $\mathcal{T}$ whose genome set is included in or equal to the genome set of S_k and (3) the weight of an edge between S_k and S_ℓ, denoted by $w(\{k, \ell\})$, is the sum, over all pairs $\{i, j\}$ such that i is a leaf of S_k and j a leaf of S_ℓ, of the number of occurrences of the cherry $\{i, j\}$ in $\mathcal{T}$. $\mathcal{L}$ can be computed initially (i.e. for $\mathcal{S} = \{\{1\}, \ldots, \{m\}\}$) from $\mathcal{T}$ in time $O(q + m^2)$ and space $O(m^2)$.

It follows from the definition of $\mathcal{L}$ that

1. $c(\mathcal{S}_{i,j}, \mathcal{T}) - c(\mathcal{S}, \mathcal{T}) = w(i) + w(j) - 2w(\{i, j\})$,
2. updating $\mathcal{L}$ after replacing $\mathcal{S}$ by $\mathcal{S}_{i,j}$ can be done by
   (a) contracting the edge between S_i and S_j to create a new vertex containing the new tree obtained by joining them,
   (b) setting the weight w of the new vertex to $w(i) + w(j) + w(\{i, j\})$ and
   (c) replacing all multiple edges adjacent to the new vertex by a single edge whose weight is the sum of the weights of these edges.

Thus, deciding which pair of trees S_i and S_j to join and updating $\mathcal{L}$ can be done in time linear in the number of edges of $\mathcal{L}$, that is in time $O(m^2)$. Procedure Completion-Leaf-Cherry performs $m - 1$ iteration of the loop in step 2 and the total time complexity is then in $O(q| + m^3)$. The space complexity is in $O(m^2)$ due to the necessity to encode the complete graph $\mathcal{L}$ that has $O(m)$ vertices.

Proof of Proposition 3

A depth-first traversal of T, in time $O(n)$, is required before applying Procedure Relabel for the initial labeling of the vertices of T by their genome sets. Finding the levels of valid vertices of T requires a second preorder tree traversal, and for each set of highest valid vertices, relabeling the vertices requires to compare their genome sets, which can be done in time proportional to the number g of different genomes. Therefore, Procedure Relabel can be done in time $O(g \times n)$.

For each level of T, Procedure Compute-Loss-Number requires to partition of the forest $\mathcal{F}$ into its subforest, which is done in time proportional to g by comparing genome sets for one level, and thus in time and space $O(g \times n)$ for all the levels of T.

Computing the phylogeny P_j during step 6 of Procedure Compute-Loss-Number can be done in time that is cubic in the number of leaves of P_j (Proposition 2), and linear in the number of considered trees of the forest (the sum of the factors q in the complexity of Procedure Completion-Leaf-Cherry over the whole process adds up into a $O(n)$ factor). We can then perform all calls to Procedure Completion-Leaf-Cherry in $O(n + g^3)$ and space $O(g^2)$, that is $O(g \times n)$.

Finally, computing $\ell(T, S)$ can be done in linear time. Therefore, the time complexity of the whole heuristic is in $O(g \times n + g^3)$ and the space complexity in $O(g \times n)$.

number of gene losses	1	2	3	4	5	6	7	8	9	10	11
number of gene families	39	238	226	203	135	75	46	23	10	4	1

Table 1. Distribution of the number of observed gene losses in the 1000 simulated gene families.

Dataset	1	2	3	4	5	6	7	8	9	10
Gene losses : observed	384	396	377	372	347	371	396	387	366	381
Gene losses : minimum	348	358	338	341	316	335	369	353	345	355
Gene losses : heuristic	348	393	338	341	373	335	369	353	395	401

Table 2. Results by dataset. Row 2 gives the exact number of gene losses observed during the generation of the gene trees. Row 3 gives the minimum number of gene losses computed by the branch-and-bound algorithm. Row 4 gives the number of gene losses inferred by our heuristic.

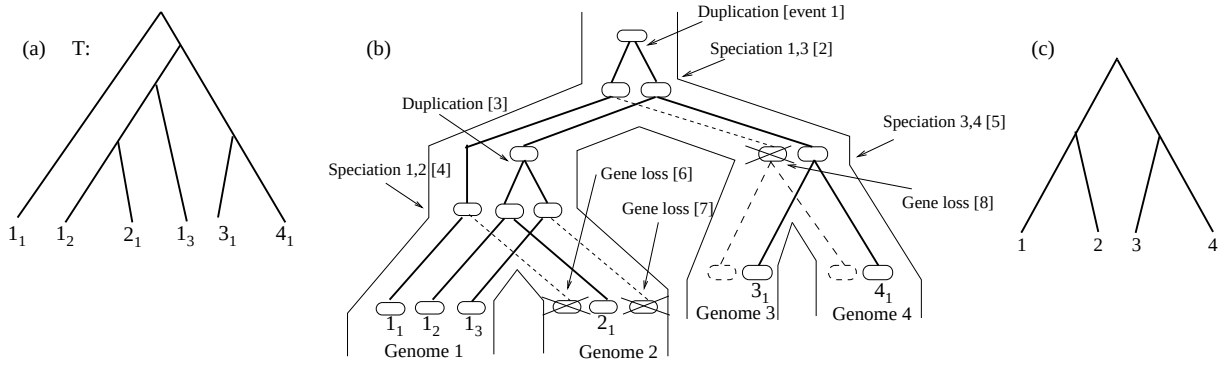


Fig. 1. (a) The gene tree T; (b) A compact representation of DLS-history explaining T where the order of events is given between brackets; the plain ovals represent extant and ancestral individual genes and dotted branches represent lost genes; the genes are denoted as k_i meaning "gene i in genome k". (c) The induced species tree S.

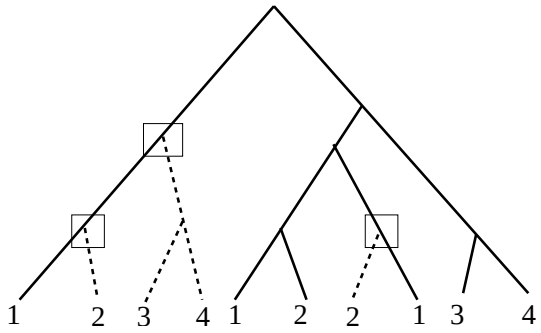


Fig. 2. Three subtree insertions are required to transform the gene tree of Figure 1.a. into a DS-tree. They correspond to the three gene losses in the history represented by Figure reffig:DLS-history.b.

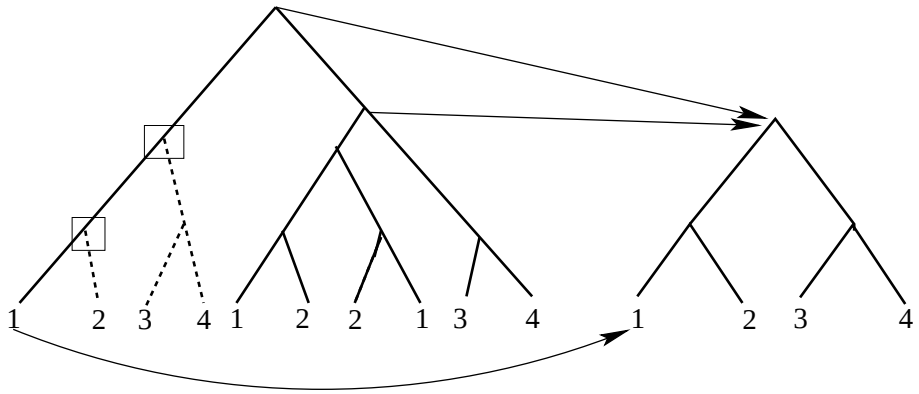


Fig. 3. Inferring subtree insertions in the reconciliation framework. The root x of the gene tree (left tree) is a duplication vertex as indicated by the three arrows that represent the LCA mapping M from the gene tree to the species tree (right tree), and the two subtree insertions on the left branch, from x to x_ℓ correspond to the $d(M(x_\ell)), M(x) + 1$ gene losses indicated by formula (1).

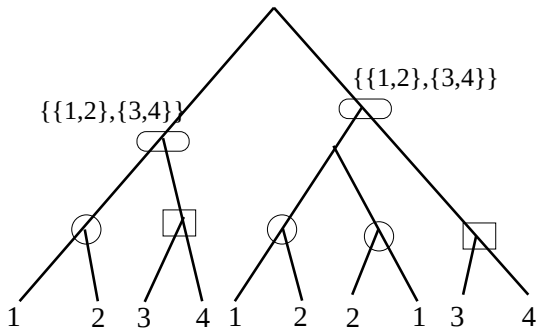


Fig. 4. Illustration of the notion related to borders: the partition associated to each valid internal vertex is shown; the border $\mathcal{B}$ of T contains the two vertices indicated by ovals, with the associated partition $\{\{1,2\},\{3,4\}\}$. The vertices indicated by circles form the border of the forest F_r containing the subtrees of T whose leaves belong to $\{1,2\}$. The ertices indicated by squares form the border of the forest F_ℓ containing the subtrees of T whose leaves belong to $\{3,4\}$.

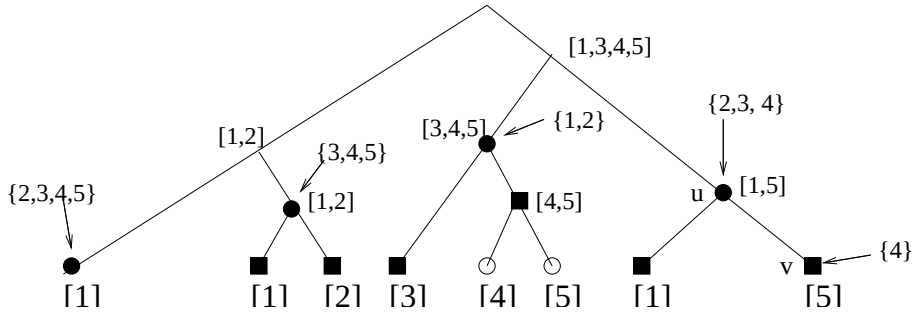


Fig. 5. An illustration of PROCEDURE RELABEL for a tree on the genome set $\mathcal{G} = \{1, 2, 3, 4, 5\}$. For each vertex x, the label in square brackets is $L(x)$ of x and the label in brackets is the genome subset inserted by PROCEDURE RELABEL to create L'. This tree has two levels: $\mathcal{V}_1$ is the set of vertices indicated by black circles and $\mathcal{V}_2$ is the set of vertices indicated by black squares. The empty circles leaves represent $\mathcal{F}$, the set of expanded leaves, so $r(T) = \{4, 5\}$.

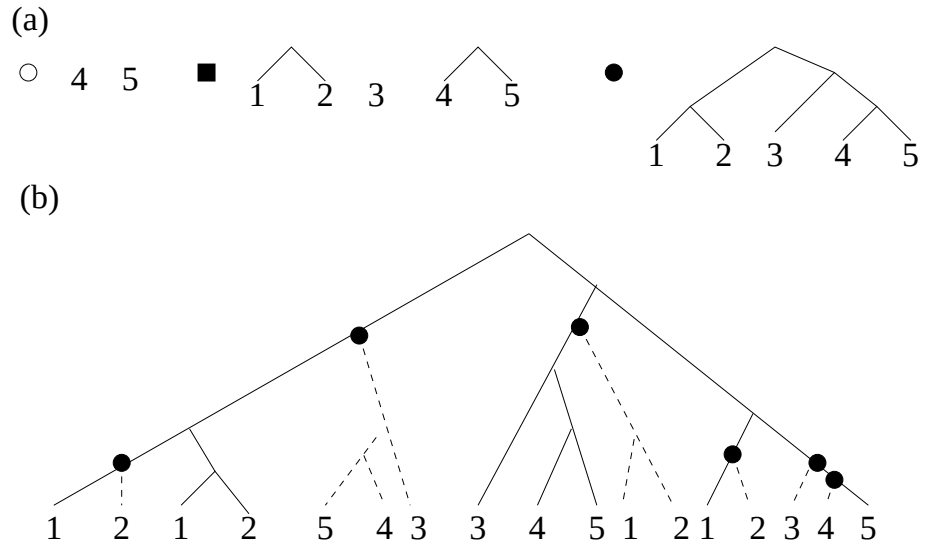


Fig. 6. Applying Procedure Compute-Loss-Number on the input (T, L') given by Fig 5. (a) The three sets of species trees computed for each of the three levels. The species tree S is the one associated to the first level. (b) The DS-tree obtained by inserting subtrees following the 6 gene loss events induced by the LCA mapping between S and T.

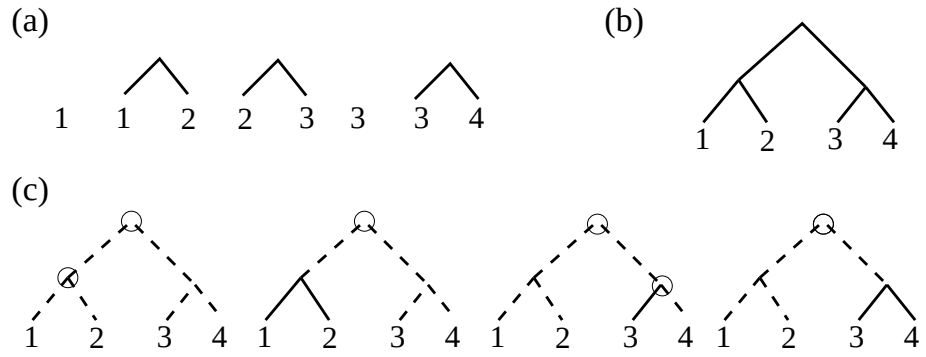


Fig. 7. Completing a set of four leaves and cherries. (a) The leaves and cherries. (b) A species tree. (c) The completion, where inserted subtrees and edges are represented with dotted lines and insertions sites are represented by empty circles. The total number of subtree insertions is six.

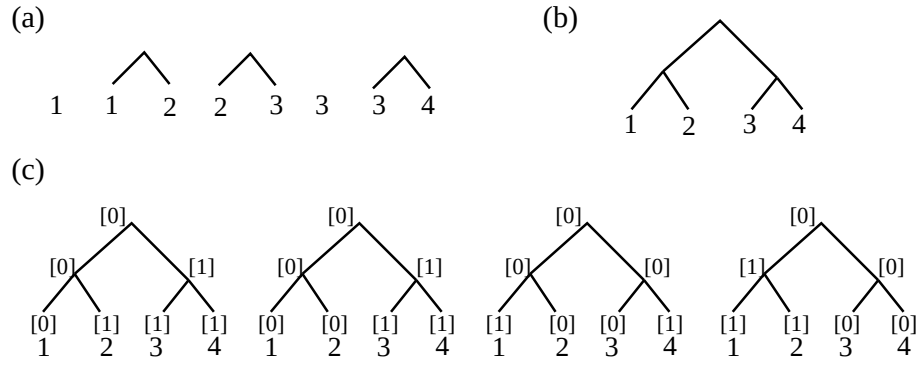


Fig. 8. Illustration of the link between completing leaves and cherries and the Camin-Sokal parsimony problem. (a) A set of leaves and cherries. (b) A species tree S. (c) The labellings of the vertices of S, represented between square brackets, according to the four leaves and cherries. There are six character gains (two for the first and third trees, ¡ and one for the second and fourth trees).

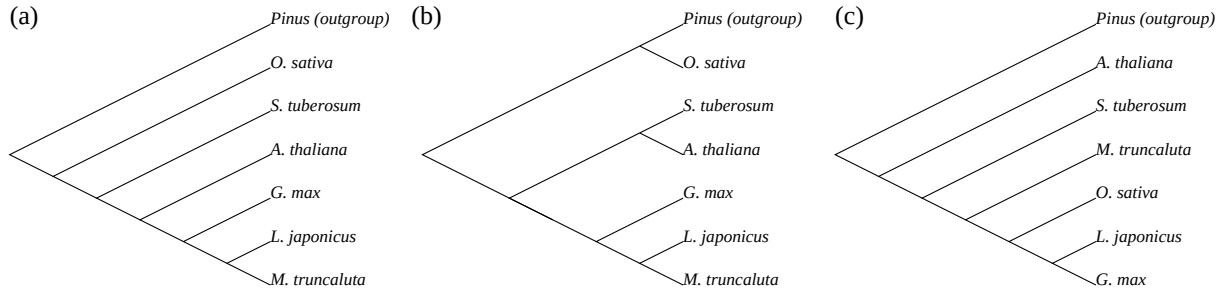


Fig. 9. Species trees for the 577 plants gene families. (a) Accepted species tree (Fig. 2 in (Sanderson and McMahon 2007)). (b) Optimal species trees obtained with the branch-and-bound algorithm. (c) Species tree obtained with the heuristic.