Inferring a duplication, speciation and loss history from a gene tree (extended abstract)*

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

[1] Department of Mathematics, Simon Fraser University, 8888 University Drive, V5A 1S6, Burnaby (BC), Canada, cedric.chauve@sfu.ca
[2] CGL and LaCIM, UQAM, Montréal, Canada
[3] DIRO, Université de Montréal, CP6128, succ. Centre-Ville, H3C 3J7, Montréal (QC),Canada, [mabrouk,doyonjea]@iro.umontreal.ca

Abstract. We consider two 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, and without gene loss. We show that this question can be answered in linear time, and that such a gene tree induces a single species tree consistent with a history with no loss. We then present a heuristic for the following problem: if a gene tree can not be explained without gene loss, what is the minimum number of losses involved in an evolutionary history of the gene family. We finally evaluate our algorithms on a dataset of plants gene families.

1 Introduction

The duplication of genetic material, from a single gene to the whole-genome, is a fundamental process in the evolution of species, and in particular eukaryotes [12, 6]. As a consequence, in most nuclear genomes, many genes are present in multiple copies, that define *gene families*. Gene families evolve, from a single ancestral gene, through microevolutionary events at the nucleotide level, and macroevolutionary events at the genomic level, such as gene duplication, gene loss, genome rearrangements, and speciation events (see [5] and references there). Understanding the evolution of gene families is a fundamental problem that has several applications. 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 following a speciation event. 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. For whole genome analysis based on gene orders and

* Work supported by grants from NSERC and SFU.

rearrangements, 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 (see [2]).

As the notion of orthology and paralogy is directly related to the history of speciation and duplication events during genomes evolution, a natural way of distinguishing between the two types of gene homologues is to infer these events from the phylogenetic tree of a gene family. This question has been widely 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 [10]. The main algorithmic approach developed to solve this problem, the gene tree/species tree *reconciliation*, allows to identify the duplications with respect to the speciation events in the species tree [4, 13]. It is based on a mapping of the gene tree into the species tree, that can be done in linear time [15, 3, 16].

Here we consider the more general case where the species tree is unknown. In this context, a natural question is to infer a species tree from a set of gene trees, that optimizes a given criterion, either combinatorial, like the number of duplications and/or losses [13, 11], or probabilistic [1]. However, we follow a different approach, as we start from a decision question, motivated among other reasons, by the importance of duplications and speciations 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 Duplication/Speciation history, we call it a DS-tree (a terminology inspired from [9]). Otherwise, we explain the non-agreement between the gene tree and any DS history by the presence of gene loss events, and we consider the problem of minimizing such number of gene losses. The more general problem of minimizing duplications and losses (in the minimum mutation cost model) for reconciling a set of gene trees has been shown to be NP-hard [13].

In Section 2, we define the notion of a DS-tree in both the frameworks of evolution and reconciliation. We then show, in Section 3, that deciding if a gene tree is a DS-tree can be answered in linear time[4], and that in such a case, there is a single species tree that is compatible with the corresponding Duplication/Speciation history. In the case where a given gene tree T is not a DS-tree, T can be derived from a DS-tree by a series of gene losses. We introduce, in Section 4, the problem of finding the

[4] For space reasons, all proofs are omitted and will appear in the full version of this paper.

minimum number of gene losses that are needed to transform a DS-tree into T, and we give an efficient heuristic for this problem running in time $O(g \times n)$, where n is the size of T and g is the number of genomes represented in T. We finally analyze, in Section 5, a dataset of plant gene families taken from [14].

2 Duplication/Speciation history and Reconciliation

Duplication/Speciation history. Let $\mathcal{G} = \{1, 2, \cdots, g\}$ be a set of integers representing g different species (genomes). A species tree for $\mathcal{G}$ is a binary tree with exactly g labeled leaves, where each $i \in \mathcal{G}$ represents 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.

We say that T is a *Duplication/Speciation tree* (or simply a DS-tree) if there exists a history involving only duplication and speciation events that can lead to the observed tree T. Hereafter, we formally define a Duplication/Speciation history (from now called DS history). See Figure 1 for an illustration.

Definition 1. Let $\mathcal{T} = (T^1, T^2, \cdots T^n)$ be an ordered sequence of n gene trees. We denote by g_k the number of genomes represented by T^k for any k. We say that $\mathcal{T}$ is a DS history if and only if:

1. $T^1 = x$ is a tree restricted to a single vertex x and $g_1 = 1$;
2. For $0 < k < n$, one of the two following situations hold:
   (a) *Duplication event:* T^{k+1} is obtained from T^k by adding two children y and z to a leaf x, and labeling them as x.
   (b) *Speciation event:* There exists i, $1 \leq i \leq g_k$, such that T^{k+1} is obtained from T^k by adding two children y and z to each leaf x of T^k labeled i, and labeling one of the two new nodes by i and the other by $g_k + 1$. Moreover, $g_{k+1} = g_k + 1$.

Let $\mathcal{T}$ be a DS-history leading to a gene tree T. Then, by construction, $\mathcal{T}$ leads to a unique species tree S induced by the speciation events (see Figure 1.a. and b.). We say that the species tree S is *DS-consistent with* T.

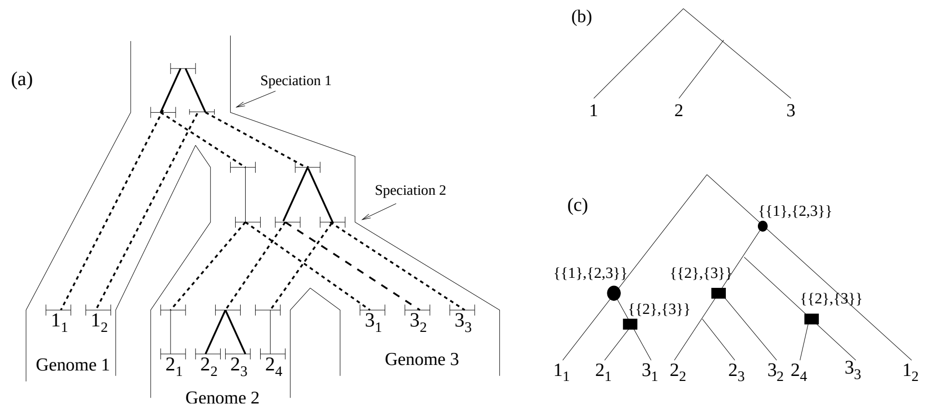


Fig. 1. (a) A DS-history; the segments represent the individual genes; the duplication events are indicated by bold lines, and the speciation events by dashed lines; the genes are denoted as k_i meaning "gene i in genome k". (b) The induced species tree S. (c) The induced gene tree T; notations introduced in Section 3: the partition associated to each internal node is shown; the border B of T contains the two nodes indicated by plain circles, with the associated partition $\{\{1\},\{2,3\}\}$, the nodes indicated by plain squares form the border of the forest F_r containing the subtrees of T whose leaves belong to $\{2,3\}$.

Reconciliation. Suppose that a species tree S is already known for $\mathcal{G}$. Then a natural question is to know whether S is DS-consistent with T. This question can be answered by using the classical reconciliation approach that "embeds" the gene tree T into the species tree S [7, 13]. The potential non-congruence between a species tree and a gene tree can then be explained by a minimum number of gene losses. More precisely, the reconciliation approach aims to infer a duplication/loss 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. We denote by $\ell(T, S)$ the number of loss events.

In this framework, the notion of DS-tree and DS-consistent species tree can be stated as follows (see [7] for a proof of the equivalence between the two approaches): a gene tree T on $\mathcal{G}$ is a DS-tree if there exists a species tree S on $\mathcal{G}$ such that $\ell(T, S) = 0$, in which case S is said to be DS-consistent with T.

3 Recognizing a DS-tree

In this section, we propose two characterizations of a DS-tree following from the fact that a DS-tree should lead to a species tree S that is DS-consistent with T. The first follows a bottom-up approach, and is the base of the linear-time recognition algorithm presented at the end of this section. The second characterization follows a top-down strategy and leads naturally to 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. Let T be a gene tree on a genome set $\mathcal{G} = \{1, \ldots, g\}$. 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. We also denote by x_l and x_r respectively the left and right child of x.

A *cherry* of T is a subset $\{i, j\}$ of $\mathcal{G}$ such that $L(x) = \{i, j\}$ for a given vertex x of T.

Definition 2. A cherry $\{i, j\}$ is said to be a *DS-valid cherry* for T if, for any vertex x_l such that $L(x_l) = \{i\}$ (resp. $\{j\}$) and $L(x) \neq \{i\}$ (resp. $\{j\}$) where x is the parent of x_l, the sibling x_r of x_l is such that $L(x_r) = \{j\}$ (resp. $\{i\}$).

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 every internal vertex x such that $L(x) = \{i, j\}$ by a leaf labeled $g + 1$.

Let x be an internal vertex of T. The unordered pair $\{L(x_l), L(x_r)\}$ is called the *partition associated to* x. We say that x *is valid* iff $L(x_l) \cap L(x_r) = \emptyset$. Let F be a forest, that is a set of one or more trees. We say that a set X of vertices of F is *covering* F iff each leaf belonging to a tree of F is a descendant of a unique vertex of the set X. We say that a vertex x is *higher* than a vertex z if z is a descendant of x. Let $B = \{b_1, \ldots, b_k\}$ be the set of highest valid vertices of a forest F: B is called a *border* iff it is covering F and all the partitions associated to the vertices of B are identical. Let B be a border of a forest F, and $\{P_l, P_r\}$ be the partition generated by the vertices of B. We denote by F_l (resp. F_r) the set of subtrees whose leaves labels belong to P_l (resp. P_r) (see Figure 1.c. for an illustration of notations).

Definition 3. A *DS-valid forest* is recursively defined as follows:

1. It is a set of leaves or
2. It has a border and its resulting forests F_l and 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 $g = 1$, 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.

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.*

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 the species tree that is DS-consistent with T, if any.

```
Algorithm DS-recognition ($T$)
1.  Let $S$ be an empty tree and $m = g + 1$
2.  Perform a depth-first traversal of $T$, and let $x$ be the current vertex
3.      IF $x$ is an internal vertex with children $x_l$ and $x_r$ such that
4.          $L(x_l) = \{i\}$ and $L(x_r) = \{j\}$ and $i \neq j$ THEN
5.          FOR EVERY vertex $z_l$ such that $L(z_l) = \{i\}$ DO
6.              Let $z_r$ be the sibling of $z_l$ and $z$ its parent
7.              IF $L(z_r) = \{j\}$ THEN replace $T_z$ by a leaf labeled $m$
8.              ELSE IF $L(z_r) \neq \{i\}$ THEN RETURN FALSE
9.          IF there remains a vertex $x$ with $L(x) = \{j\}$ THEN
10.             RETURN FALSE
11.         Add to $S$ a subtree with root labeled $m$ and children labeled $i$ and $j$
12.         Increment $m$
13. RETURN $S$
```

Theorem 2. *Given a gene tree T with n vertices, Algorithm DS-recognition returns FALSE iff 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.*

4 Inferring gene losses in a non DS-tree

4.1 Problem statements

If a gene tree T is not a DS-tree, and assuming that the given gene tree T is correct (see [8] for a discussion on the case where gene duplications can lead to an incorrect gene tree for a gene family), this implies that some homologous genes are missing or have been deleted or transformed to pseudo-genes during evolution. When a species tree S is known, the reconciliation method can be used to infer a scenario of minimum number $\ell(T,S)$ of gene losses that has led to the observed tree. In this section, we assume that the species tree is unknown, and consider the following natural optimization problem.

Duplication/Loss problem: Given a gene tree T that is not a DS-tree, find a species tree S such that $\ell(T,S)$ is minimum.

This problem can be related to those considered in [13] that compute, for a given gene tree T (or more generally a set of gene trees $T_1,\ldots,T_k$), a species tree S minimizing the total number of duplications (in the so-called duplication cost model) or duplications and losses (in the mutation cost model). They have both been shown to be NP-hard [13], but fixed-parameter tractable [11].

We will instead consider an equivalent formulation of this problem, based on the following property: if T is not a DS-tree, then for every species tree S, there is a DS-tree T^S that can be obtained from T by inserting a minimum number of subtrees such that S is DS-consistent with T^S. Each of these *subtree insertions* represents a gene loss in a given ancestral or extent genome. This way to relate T to a DS-tree T^S, for a given species tree S, leads to the following optimization problem, in the case of an unknown species tree:

Subtrees Insertion Problem: Given a gene tree T that is not a DS-tree, find the minimal number δ of subtree insertions in T allowing to transform T into a DS-tree T'. We denote by (S,δ) a solution to this problem, where S is such that $T' = T^S$.

It follows from [7] that:

Proposition 1. *The Duplication/Loss Problem and the Subtrees Insertion Problem are equivalent: a species tree S is a solution to the Duplication/Loss Problem with $\ell(T,S)=\delta$ if and only if (S,δ) is a solution to the Subtrees Insertion Problem.*

4.2 A heuristic for the Subtrees Insertion Problem

We now describe an algorithm allowing to obtain an upper bound on the minimum number of subtrees insertions – called *insertions* from now for short – required to transform a gene tree T into a DS-tree.

The method can be decomposed in three steps: (1) recursively label the vertices of T with subsets of the genome set $\mathcal{G}$, (2) use these labels to construct a DS-tree from T, and (3) factorize some of the insertions to reduce the total number of insertions.

Labeling the vertices of T. Initially, each vertex x of T is labeled by its genome set $L(x)$. A set of vertices is said *consistent* if and only if any two vertex labels with a non-empty intersection are identical. *Procedure Relabel* below then relabels the vertices of T in order to obtain successive levels of consistent vertices. It uses the following concepts: a set $\{x_1, \ldots, x_k\}$ of vertices is said to be *connected* if the intersection graph induced by the labels of these vertices (the nodes of the graph are the x_i's and two nodes are connected if their labels have a non-empty intersection) is connected. *Completing* the labels of a set $\{x_1, \ldots, x_k\}$ of connected vertices consists in adding to the label of every vertex x the subset $\cup_{i=1}^{k} L(x_i) \backslash L(x)$ of $\mathcal{G}$, in order that its new label is $\cup_{i=1}^{k} L(x_i)$ (see Fig. 2).

```
Procedure Relabel (T)
1.  $F$ is the forest restricted to the tree $T$;
2.  WHILE $F$ is not restricted to a set of leaves DO
3.      Let $\mathcal{V}$ be the set of highest valid vertices of $F$;
4.      Complete the labels of every maximal connected subset of $\mathcal{V}$
5.      Let $F$ be the forest of $\mathcal{V}$;
6.  IF $g$ is inserted in the labels of a vertex $x$ and of a descendant of $x$ THEN
7.      Remove $g$ from the label of $x$.
```

The successive sets of highest valid vertices of T considered in *Procedure Relabel* are called the successive *levels of T*. An illustration of this procedure is given in Figure 2.

A first transformation of T into a DS-tree. We now describe how to use the vertices labels computed by *Procedure Relabel (T)* in order to insert subtrees into T, in such a way that the result is a DS-tree. We denote by L the new labeling of the vertices of T computed by *Procedure Relabel (T)*: for a vertex x of T, $L(x)$ is the new genome set associated to x. 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 extend the notion of connected subset of the $x_i's$ used in *Procedure Relabel* as follows: a subset of the $T_i's$ is said to be connected if the intersection graph induced by the labels of the corresponding $x_i's$ is connected. We call the *partition of $\mathcal{F}$ by genome sets* the unique partition of $\mathcal{F}$ into forests defined as maximal connected subsets of the $T_i's$.

The construction algorithm is described below, and illustrated in Figure 3.

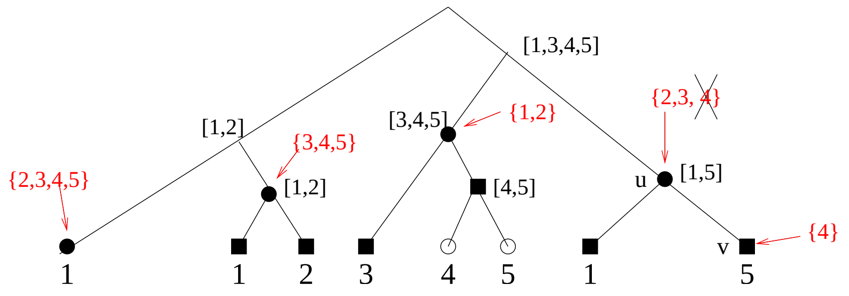


Fig. 2. An illustration of *Procedure Relabel* for a tree on the genome set $\mathcal{G} = \{1, 2, 3, 4, 5\}$. For each internal vertex x, the label in square brackets is the genome set $L(x)$ of x and the label in brackets is the genome subset inserted by Procedure Relabel. This tree has three levels: the first level is the set of vertices indicated by bold circles, the second is the set of bold square vertices, and the third is the two leaves indicated by white circles. The crossed genome in the label of vertex u is the genome removed after applying instruction 7 of *Procedure Relabel*.

```
Procedure Construct-DSTree (T,L)
1.   FOR each level of $T$ (involving insertions) beginning with the last level DO
2.       Let $\mathcal{F}$ be the forest representing the current level;
3.       Let $F_1, \cdots F_p$ be the partition of $\mathcal{F}$ by genome sets;
4.       FOR each subforest $F_i$ DO
5.           Let $\mathcal{G}_i$ be the genome set of $F_i$;
6.           Let $P$ be an arbitrary phylogeny for $\mathcal{G}_i$;
7.           FOR each vertex $x$ of $T$ such that $L(x) \subset \mathcal{G}_i$ DO
8.               Perform the unique set of subtrees insertions,
9.               in the subtree $T_x$ and on the edge from $x$ to its parent
10.              leading to the phylogeny $P$
```

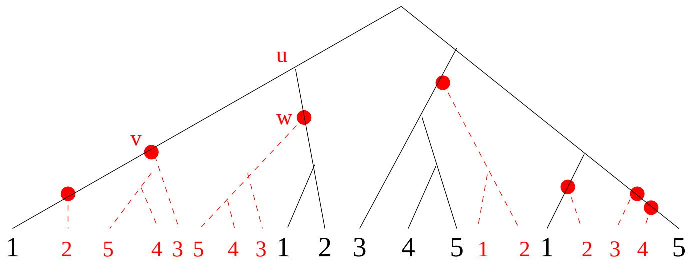


Fig. 3. The result of applying *Procedure Construct-DSTree* on the input (T, L) given by Fig 2. The inserted branches are indicated by dotted lines.

The construction procedure can be reformulated as follows: consider successively each level $\mathcal{F}$ of T beginning with the last one, for each tree T_i of $\mathcal{F}$ rooted at x_i, perform the subtrees insertions leading to the genome set $L(x_i)$, and then replace each tree of T_i by a single leaf. The key observation is that each level considered by this procedure consists solely of leaves and cherries. Therefore, for any x_i, any arbitrary phylogeny P representing the genome set $L(x_i)$ can be obtained by subtrees insertions in T_i. In other words, at each level, there is a coherent way of inserting missing genes in a way leading to the same phylogeny at each node of the tree. This is the main argument used in the proof of the following theorem.

Theorem 3. *Let T be a gene tree that is not a DS-tree, and (T, L) the output of* Procedure Relabel(T). *The gene tree computed by* Procedure Construct-DSTree(T,L) *is a DS-tree.*

Corollary 2. *The number of insertions performed by* Procedure Construct-DSTree *is an upper bound on the minimum number of insertions necessary to transform T into a DS-tree.*

Note that in step 6 of *Procedure Construct-DSTree*, we choose an arbitrary phylogeny for the considered subset of taxa. This point could be improved as this phylogeny can be non optimal in terms of the number of subtrees insertions. It could be approached, for example, in a greedy way by selecting the phylogeny that induces the minimum number of subtrees insertions.

Reducing the number of subtrees insertions. A further improved upper bound can be obtained by "factorizing" the subtree insertions made by *Procedure Construct-DSTree*. Let T' be the tree computed by *Procedure Construct-DSTree*, u be an internal vertex of T' that is also a vertex of T. Let $L(u_l)$ be the left genome set of u in T and $L(u_r)$ be the right genome set of u in T. Suppose that the two children v and w of u in S are two inserted vertices, and let $(v, L(v_r))$ and $(w, L(w_l))$ be the two inserted branches. Then we perform the following modification of T':

1. If $L_u = R_u$, then remove the two branches $(v, L(v_r))$ and $(w, L(w_l))$ and insert the subtree T_{w_l} on the branch from u to its parent (see (1) of Figure 4).
2. If $L(v_r) = L(w_l)$ and $L(v_l) = L(w_r)$, then remove the vertices v and w and the subtrees T'_{v_r} and T'_{w_l} (see (2) of Figure 4).

Applying the factorization rule (1) on the tree of Figure 3 gives rise to the tree of Figure 5, leading to 6 subtrees insertions.

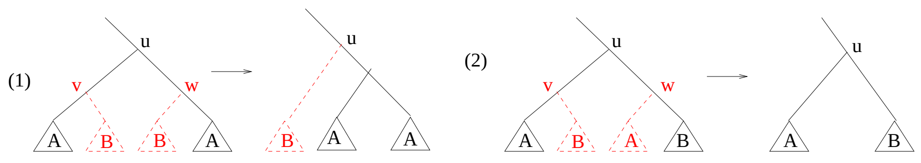


Fig. 4. Illustration of the two factorization rules.

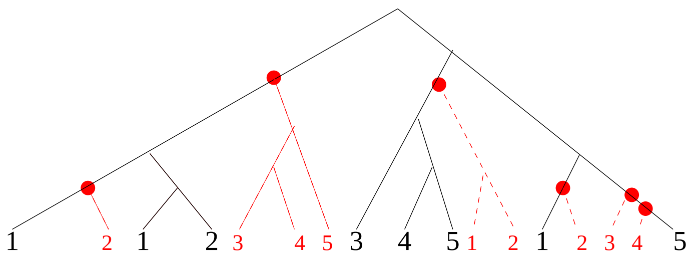


Fig. 5. The result of applying the factorization rules on the DS-tree of Figure 3.

Complexity. Let n be the size (number of vertices) of T. A depth-first traversal of T, in time $O(n)$, is required before applying *Procedure Relabel* for the initial labeling of T's vertices by their genome sets. Finding the sets of highest valid vertices then 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 $\mathcal{F}$ of T, *Procedure Consruct-DSTree* requires to partition $\mathcal{F}$ into its subforest, which is done in time proportional to g by comparing genome sets for one level, and thus in time $O(g \times n)$ for all the levels of T. On the other hand, as each tree insertion can be done in constant time, and a maximum of g tree insertions are performed at each vertex of T, the time complexity for tree insertions is in $O(g \times n)$. Therefore, *Procedure Consruct-DSTree* can be done in time $O(g \times n)$. Finally, the step of reducing the number of subtree insertions can be done in time $O(n)$, by performing a depth-first traversal of T. Therefore, the time complexity of the whole algorithm is in $O(g \times n)$.

5 Experimental results

We describe the results obtained with the algorithms presented in the previous sections on the 577 gene families studied in [14], in a study of the phylogeny of seven angiosperm genomes from EST data. We focus mainly, in this preliminary experiment, on the computational properties of our algorithms, and in particular on the quality of our heuristic for the Duplication/Loss Problem.

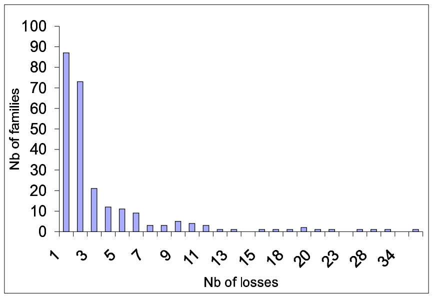
Data. Each of the 577 gene families contains at least four genes and spans at least three genomes. The gene trees were obtained with PAUP, using a maximum likelihood approach (see [14] for a detailed description of the process followed to obtain these gene families and gene trees). The data, including the gene trees and statistics on the size of 577 gene families, and the results of our experiments are available on a companion website, accessible at `http//www.lacim.uqam.ca/~chauve/CG07`.

Results. First, we found that 333 of the 577 gene trees are DS-trees. However, without surprise, most of these families exhibit few gene duplications. For example, 89 of these 333 families contain 4 genes and span 3 species, while only 7 of the 59 gene trees that span the 7 species are DS-trees (see the file `STATS.txt` on the companion website for the complete statistics). Next, we applied to the 244 remaining gene trees (called non-DS trees from now) the heuristic described in Section 4 to compute an upper bound on the minimum number of gene losses needed to explain the observed gene tree. The results are summarized in the left graphics of Figure 6, and show that many gene families can be explained with few gene losses. We also implemented a branch-and-bound algorithm (to be described in the full version of this paper) in order to assess, on this particular dataset, the quality of our heuristic. The results, summarized in the right graphics of Figure 6, show that it performs well, as for 214 gene trees, it computed the optimal number of gene losses.

Finally, our branch-and-bound algorithm allowed us to compute, for each gene tree that is not a DS-tree, the number of species trees that induce a minimum number of gene losses. As shown in the table below, in most cases, there is a unique optimal species tree.

Number of species trees:	1	2	3	4	5	6	7	13	15
Number of families:	179	16	34	2	6	3	1	2	1

Table 1. Distribution of the 244 gene families with a non-DS tree (second line) according to the number of species trees inducing a minimum number of gene losses (first line).



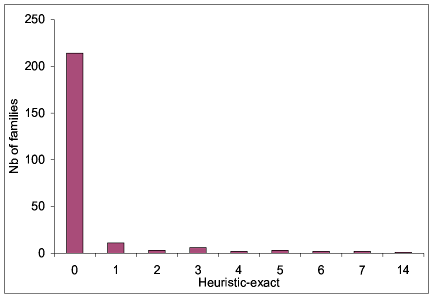


Fig. 6. Left: Distribution of the 244 gene families having a non-DS tree (y axis) according to the number of gene losses inferred by our heuristic (x axis); Right: Distribution of the 244 gene families having a non-DS-tree (y axis) according to the difference between the minimum number of losses needed to transform them into a DS-tree, and the upper bound obtained by our heuristic (x axis).

Discussion. On the considered dataset, we found that many gene families could be explained with few gene losses. However, as these gene families are obtained from EST data, it is very likely that many of them are incomplete and it should be expected that more gene losses are required to explain the true evolution of these families. The distribution of the size of the gene families that can be explained without gene losses illustrates this point, as most of them are quite small, while most gene families with many genes and/or genomes require significantly more gene losses.

From an algorithmic point of view, these experiments suggest that our heuristic works well and that, together with our branch-and-bound algorithm, it gives an efficient way to compute the minimum number of gene losses to explain the evolution of a gene family and the corresponding species tree(s). As a consequence, our approach seems to be an interesting candidate to propose quickly, for a given set of gene trees, a set of species trees (for example the species trees inferred from gene trees that can be explained with few gene losses) that can be analyzed using tree consensus or supertree methods.

6 Conclusion

We proposed in this paper a study of gene trees with a focus on duplication and speciation events. In particular, we showed that deciding if a gene tree is a DS-tree is not difficult, and lead to a single species tree. We also introduced a new way to study the evolution of a gene family by minimizing the number of gene losses. Our preliminary experimental results suggest that our approach is worth further studies, and should be compared with the two other reconciliation approaches based on minimizing the number of duplications and the number of duplications and losses.

Among the algorithmic open problems that our work suggest, the most natural is the complexity of the Subtrees Insertion Problem. In a different perspective, preliminary results on yeast gene families show that our approach needs to be generalized to non fully resolved gene trees (work in progress). It would also be very useful to consider not only gene losses to complete gene trees, but also tree rearrangement operations that could account for potential errors in the obtained gene trees. Finally, as our approach relies on inferring a (or a set of) species tree(s) for each gene family, it would be useful to measure the significance of such species trees.

References

1. L. Arvestad, A.-C. Berglung, J. Lagergren and B. Sennblad. Gene tree reconstruction and orthology analysis based on an integrated model for duplications and sequence evolution. *RECOMB 2004*, p. 326–335. 2004.
2. G. Blin, C. Chauve, G. Fertin, R. Rizzi and S. Vialette. Comparing genomes with duplications: a computational complexity point of view. To appear in *IEEE/ACM Trans. on Comput. Biol. and Bioinformatics*. 2007.
3. K. Chen, D. Durand and M. Farach-Colton. NOTUNG: a program for dating gene duplications and optimizing gene family trees. *J. Comput. Biol.*, 7(3-4):429-444. 2000.
4. J.A. Cotton and R.D.M. Page. Going nuclear: gene family evolution and vertebrate phylogeny reconcilied. *Proc. R. Soc. Lond. B*, 269:1555-1561. 2002.
5. D. Durand, B.V. Haldórsson and D. Vernot. A hybrid micro-macroevolutionary approach to gene tree reconstruction. *J. Comput. Biol.*, 13(2):320–3354. 2006.
6. E.E. Eichler and D. Sankoff. Structural dynamics of eukaryotic chromosome evolution. *Science*, 301(5634):793–797. 2003.
7. O. Eulenstein, B. Mirkin and M. Vingron. Comparison of annotating duplication, tree mapping, and copying as methods to compare gene trees with species trees. In *Mathematical hierarchies and biology*, vol. 37 of *DIMACS Ser. Discrete Math. Theoret. Comput. Sci.*, p.71–93. Amer. Math. Soc., 1997.
8. M.A. Fares, K.P. Byrne and K.H. Wolfe. Rate asymmetry after genome duplication causes substantial long-branch attraction artifacts in the phylogeny of Saccharomyces species. *Mol. Biol. Evol.*, 23(2):245–253. 2006.
9. P. Gorecki and J. Tiutyn. DLS-trees: a model of evolutionary scenarios. *Theoretical Comput. Sci.*, 359(1–3):378–399. 2006.
10. R. Guigó, I. Muchnik and T.F. Smith. Reconstruction of ancient phylogenies. *Mol. Phylogenet. Evol.* 6(2):189–213. 1996.
11. M.T. Hallett and J. Lagergren. New algorithms for the duplication-loss model. *RECOMB 2000*, p. 138–146. 1996.

12. M. Lynch and J.S. Conery. The evolutionary fate and consequences of duplicate genes. *Science*, 290(5494):1151–1155. 2000.
13. B. Ma, M. Li and L. Zhang. From gene trees to species trees. *SIAM J. Comput.*, 30(3):729–752. 2000.
14. M.J. Sanderson and M.M. McMahon. Inferring angiosperm phylogeny from EST data with widespread gene duplication. *BMC Evol. Biol.*, 7(Suppl 1):S3. 2007.
15. C.M. Zmasek and S.R. Eddy. A simple algorithm to infer gene duplication and speciation events on a gene tree. *Bioinformatics*, 17(9):821–828. 2001.
16. L. Zhang. On a Mirkin-Muchnik-Smith conjecture for comparing molecular phylogenies. *J. Comput. Biol.*, 4(2):177–187. 1997.