Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 322 (2016) 35–50

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Correcting Gene Trees by Leaf Insertions: Complexity and Approximation

Stefano Beretta[1]

DISCo
Università di Milano-Bicocca
Milano, Italy

Riccardo Dondi[2]

Dipartimento di Scienze Umane e Sociali
Università di Bergamo
Bergamo, Italy

Abstract

Gene tree correction has recently gained interest in phylogenomics, as it gives insights in understanding the evolution of gene families. Following some recent approaches based on leaf edit operations, we consider a variant of the problem where a gene tree is corrected by inserting leaves with labels in a multiset M. We show that the problem of deciding whether a gene tree can be corrected by inserting leaves with labels in M is NP-complete. Then, we consider an optimization variant of the problem that asks for the correction of a gene tree with leaves labeled by a multiset M', with $M' \supseteq M$, having minimum size. For this optimization variant of the problem, we present a factor 2 approximation algorithm.

Keywords: Computational biology, Phylogenomics, Gene tree corrections, Gene Tree-Species Tree Reconciliation, Algorithms, Computational Complexity.

1 Introduction

The understanding of genome evolution is related to the identification of which evolutionary events (mainly speciations, duplications and losses, in some models lateral gene transfers) lead to the evolution of a genome [23,16]. The evolution of a *gene family* (a set of genes that originate through duplications from an ancestral gene) for a given set of species is usually represented by a *gene tree*. Once a gene tree is computed, usually via methods that rely on sequence similarity, it is compared with a species tree (a tree that represents the evolution of the set of species analysed)

[1] Email: stefano.beretta@disco.unimib.it

[2] Email: riccardo.dondi@unibg.it

http://dx.doi.org/10.1016/j.entcs.2016.03.004
1571-0661/© 2016 The Authors. Published by Elsevier B.V.

in order to identify which evolutionary events occurred in the evolution of the considered gene family [24,26]. Species trees are phylogenetic trees that are based only on speciation events, thus the evolutionary histories represented by a gene tree and a species tree can be different. The *reconciliation* of a gene tree and a species tree [7,8,9,18,25,27,31,14,3] compares the two trees, in order to infer the evolutionary events represented in the gene tree.

A related problem is the inference of the species tree, starting from a set of potentially discordant gene trees. This problem has been intensively studied under different models (see for example [10,22,2,30]) and it is known to be intractable [6,22,12,4,20].

One of the main drawbacks of reconciliation is that gene trees usually contain errors that alter the resulting evolutionary scenario [19,28]. Thus several approaches [11,15,17,29,13,21,5] have been proposed to correct gene trees before the reconciliation.

In this paper, we consider an approach that aims to remove a special kind of duplications, called *Non-Apparent Duplications* (NAD). NAD nodes can be related to errors in the gene trees [10,29], since they represent a disagreement between a gene tree and a species tree that is not directly related to a gene duplication. Thus, some recent approaches to gene tree correction aim to modify the structure of a given gene tree so that it does not contain NAD nodes, via polytomy refinement [21] or by edit operations (removal and modification) on misplaced leaves/labels [29,13,5]. More precisely, the approaches considered in [29,5] introduced two edit operations on leaves (leaf deletion and leaf modification). Here, following a similar approach, we introduce a third edit operation on leaves, *leaf insertion*, and we consider a combinatorial problem, called LeafIns, that aims to remove NAD nodes by inserting leaves associated with a given multiset M of labels. The multiset M represents a set of candidate missing leaves in the gene tree, due to errors in the reconstruction process. We consider the computational complexity of the LeafIns problem, and we show in Section 3 that it is NP-complete. Then, we consider a natural optimization version of this problem, called MinLeafIns, that aims at correcting a given gene tree by inserting the minimum number of leaves labeled by a multiset $M' \supseteq M$. For this optimization problem (which is NP-hard by the previous result), we give in Section 4 a polynomial time approximation algorithm of factor 2.

The paper is organized as follows. In Section 2, we give some preliminary definitions and properties of gene trees and species trees, and we formally introduce the two combinatorial problems we are interested in. In Section 3, we show that the LeafIns problem is NP-complete, while in Section 4 we give an approximation algorithm of factor 2 for MinLeafIns. Finally, we conclude the paper with some open problems.

2 Preliminaries

In this section, first we introduce some preliminary concepts, and we give the formal definitions of the two combinatorial problems we are interested in.

Let $\Lambda = \{l_1, l_2, \ldots, l_m\}$ be a set of labels, where each label represents a different

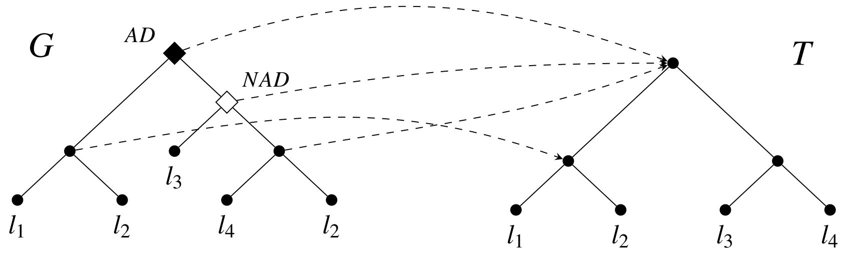

Fig. 1. A gene tree G and a species tree T. A dashed arrow from an internal node of G to an internal node of T represents its lca mapping.

species. Given a generic rooted tree U, we denote by $L(U)$ the set of its leaves, by $\Lambda(U)$ the set of labels associated with $L(U)$, and by $r(U)$ the root of U. Consider an internal node y of tree U, we denote by y_l (y_r, respectively) the left child (the right child, respectively) of y; y_r and y_l are called *sibling*. Moreover, $U[y]$ denotes the subtree of U rooted at node y. As a consequence $\Lambda(U[y])$ denotes the set of labels associated with leaves of $U[y]$. A node x on the (unique) path that connects the root of U to a node y of U is called an *ancestor* of y; y is called a descendant of x. The *parent* z of y is the ancestor of y such that (z, y) is an arc of U.

Given an ordered set $L = \langle l_1, l_2, \dots, l_p\rangle$, a *caterpillar* U on L is a binary rooted tree with p leaves, each one associated with a distinct label of L, and internal nodes $r(U), v_1, \dots, v_{p-2}$ such that: (1) there exists a path that connects $r(U), v_{p-2}, \dots, v_1$ (in this order); (2) node v_1 is adjacent to exactly two leaves, labeled by l_1, l_2, node v_i, with $1 \leq i \leq p-2$, is adjacent to a single leaf, labeled by l_{i+1}, $r(U)$ is adjacent to a single leaf, labeled by l_p.

Next, we consider two kinds of (rooted) binary trees leaf-labeled by Λ (that is each leaf of such trees is associated with a label in Λ): *species trees* and *gene trees* (see Fig. 1). The two kinds of trees differ for the allowed leaf labeling. In a species tree T labeled by Λ, each label in Λ is associated with exactly one leaf. In a gene tree G labeled by Λ, each label in Λ can be associated with at least one leaf.

An *insertion* of a leaf x with a label l in an arc $e = (u, v)$ of a gene tree G consists of: (1) defining a new node w of G, (2) removing arc e and (3) adding the three arcs (u, w), (w, v), (w, x).

In the comparison of a gene tree G and species tree T (both leaf-labeled by Λ), we consider the well-known *LCA mapping* (Least Common Ancestor mapping), denoted by $\text{lca}_{G,T}$ (see Fig. 1). Given a node x of G, $\text{lca}_{G,T}(x) = y$, where y is the farthest node of T from the root such that $\Lambda(T[y]) \supseteq \Lambda(G[x])$.

Given a species tree T, a node x of G is defined a *duplication node*, when there exists a node $z \in \{x_l, x_r\}$ (a child of x), such that $\text{lca}_{G,T}(x) = \text{lca}_{G,T}(z)$. In this case we say that a duplication occurs in x. A node of G, which is not a duplication node, is called a *speciation* node. Moreover, we can distinguish two kinds of duplication nodes (see Fig. 1 for an example of the two kind of duplications).

A duplication node x is an *Apparent Duplication node* (*AD node*), if $\Lambda(G[x_l]) \cap \Lambda(G[x_r]) \neq \emptyset$. Notice that an AD node is a duplication node for any species tree. If a duplication node is not an AD node, it is called a *NAD node*. A gene tree G

is said to be *consistent* with a species tree T when it does not contain NAD nodes (hence each node is either a speciation or an AD node).

As observed in [10,29], NAD nodes are considered to be related to errors in the gene tree. Here, we consider two combinatorial problems that aim to modify the gene tree by inserting leaves. First, we consider a decision problem that asks if a gene tree can be made consistent by inserting leaves having labels in a multiset M.

Problem 1 ***Leaf Insertion Problem****[LeafIns]*
Input: *A gene tree G and a species tree T, both leaf-labeled by Λ, a multiset M of labels in Λ.*
Output: *A gene tree G^* consistent with T such that G^* is obtained from G by inserting $|M|$ leaves each one labeled with a label in M.*

Moreover, we consider an optimization problem that asks if a gene tree can be made consistent by inserting leaves labeled by a minimum multiset M' that contains M. In fact, since LeafIns will be shown to be NP-complete, we define an optimization problem in which the constraint on the cardinality of the label multiset is relaxed.

Problem 2 ***Minimum Leaf Insertion Problem****[MinLeafIns]*
Input: *A gene tree G and a species tree T, both leaf-labeled by Λ, a multiset M in Λ.*
Output: *A gene tree G^* consistent with T such that G^* is obtained from G by inserting leaves labeled by a multiset M' of labels in Λ, with $M' \supseteq M$, and $|M'|$ is minimum.*

We conclude this section proving some properties on NAD nodes.

Lemma 2.1 *Consider a gene tree G and a species tree T, and let x be a NAD node in G. Then, in a gene tree G^* obtained by inserting leaves in G, x is either a NAD node or an AD node.*

Proof. Since x is a NAD node, it follows that at least one of its children, w.l.o.g. x_r is mapped to $y = \text{lca}_{G,T}(x)$. It follows that each leaf insertion in $G[x_r]$, does not change x to a speciation node. Moreover, notice that x_l is mapped to a node of the subtree $T[y]$.

Now, consider some leaves inserted in $G^*[x_l]$ and in the arc (x, x_r) of G, so that x_r^* and x_l^* are the children of x in G^*. Let $z = \text{lca}_{G^*,T}(x)$, where z is an ancestor of y. It follows that at least one of x_r^* and x_l^* is mapped to z, thus making x either a NAD or an AD node. Indeed, by construction, x_r^* is mapped to node on the path from y to z, while x_l^* is mapped to a node of $T[y]$ or to a node on the path from y to z. If both x_r^* and x_l^* are mapped to a proper descendant of z, then the same property holds for x and this concludes the proof. □

A consequence of Lemma 2.1 is that a NAD node of a gene tree G can only be transformed to an AD node of a consistent gene tree G^* obtained from G by leaf insertions.

Lemma 2.2 *Consider a gene tree G and a species tree T, and let x be a NAD node in G. Then, for each $p \in \Lambda(G[x])$, it is possible to make x an AD node by inserting* *a leaf labeled by p in $G[x]$, so that $G[x]$ is consistent with T.*

Proof. Let x be a NAD node of G. Consider the subtrees $G[x_l]$ and $G[x_r]$ and let p be a label in $\Lambda(G[x])$. Since x is a NAD node, it holds $\Lambda(G[x_r]) \cap \Lambda(G[x_l]) = \emptyset$. Consider w.l.o.g. a leaf g of $\Lambda(G[x_r])$ labeled by p. We insert a leaf having label p in $G[x_l]$, so that the internal node added by the leaf insertion is not a NAD node, while x becomes an AD node.

Consider the leaf f of T labeled by p. First, assume that there exists an internal node y of $G[x_l]$, farthest from the root, which is mapped by $\text{lca}_{G,T}$ to a node z of T which is an ancestor of f. Since y is a node farthest from the root, it follows that the children of y are not mapped to z. Moreover, assume w.l.o.g. that $p \in \Lambda(T[z_l])$; it follows that, since y is mapped to z, at least one of y_r and y_l (w.l.o.g. y_l) is mapped to a proper descendant of z_l, while the other node (w.l.o.g. y_r) is mapped to a descendant of z_r. Consider the insertion of a leaf labeled by p in the arc (y, y_l) and let w be the internal node introduced by the leaf insertion. Then w is a speciation node, since it is mapped to z_l, without modifying the mapping of y_l, y_r and y. Node y becomes an AD node, due to the insertion of a leaf labeled by p in $G[x_l]$.

Assume that there is no node y of $G[x_l]$ mapped by $\text{lca}_{G,T}$ to a node z of T which is an ancestor of f. Then, consider the insertion of a leaf labeled by p in arc (x, x_l). The internal node w introduced by the leaf insertion is a speciation node, since it is mapped by $\text{lca}_{G,T}$ to an ancestor u of z, such that $f \in \Lambda(T[u])$, while x_l is mapped to z. Moreover, by construction the mappings of x_l, x_r are not modified, and x becomes an AD node. □

3 Computational Complexity of LeafIns

In this section, we consider the computational complexity of LeafIns and we show that it is NP-complete. We prove this result by giving a reduction from Minimum Vertex Cover (MinVC) on Cubic Graphs. Given a cubic graph [3] $\mathcal{G} = (V, E)$, where $|V| = n$ and $|E| = m$, *MinVC on Cubic Graphs* asks if there exists a cover of $\mathcal{G}$ having size at most k, that is if there exists a subset $V' \subseteq V$, with $|V'| \leq k$, such that for each edge $\{u, v\} \in E$, at least one of u, v is in V'. Notice that obviously $k \leq n$. Moreover, we can assume that $3k - m \geq 0$. Indeed, each vertex in a vertex cover V' covers at most three edges and each edge must be covered by a vertex in V', hence $3k - m \geq 0$.

We start by defining the instance of LeafIns, that is the two subtrees G and T, and the multiset M.

The tree T (see Fig. 2) consists of a path that connects the root $r(T)$ to nodes $s_n, \ldots, s_1$ that is, one for each vertex of G; each node s_i, with $1 \leq i \leq n$, is connected to a subtree $T(v_i)$, and the root $r(T)$ is connected to a subtree $T(\gamma)$.

Each subtree $T(v_i)$, with $1 \leq i \leq n$, consists of two subtrees: a left subtree, which is a caterpillar on the ordered set $\langle \alpha_i, \beta_i, x_{i,3}, x_{i,4} \rangle$, and a right subtree, which has exactly two leaves labeled by $x_{i,1}$, $x_{i,2}$.

[3] We recall that in a cubic graph every node has degree three.

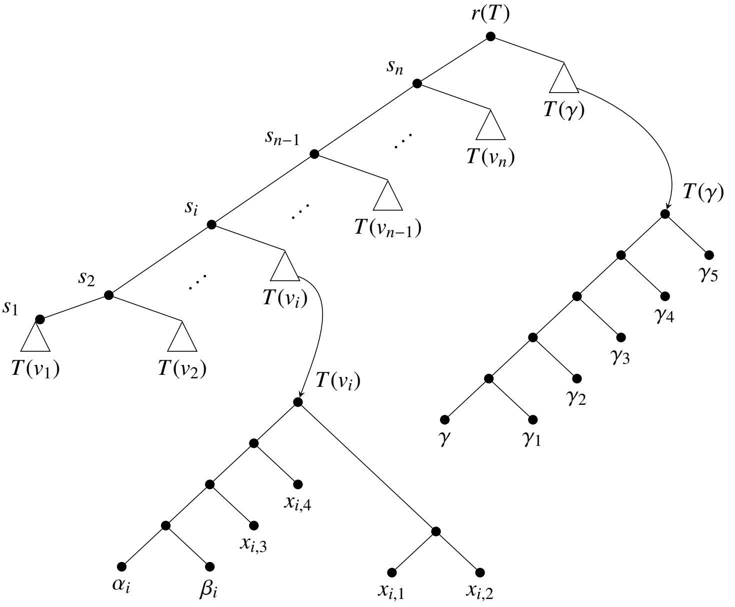

Fig. 2. The species tree T associated with an instance of *MinVC on Cubic Graphs*.

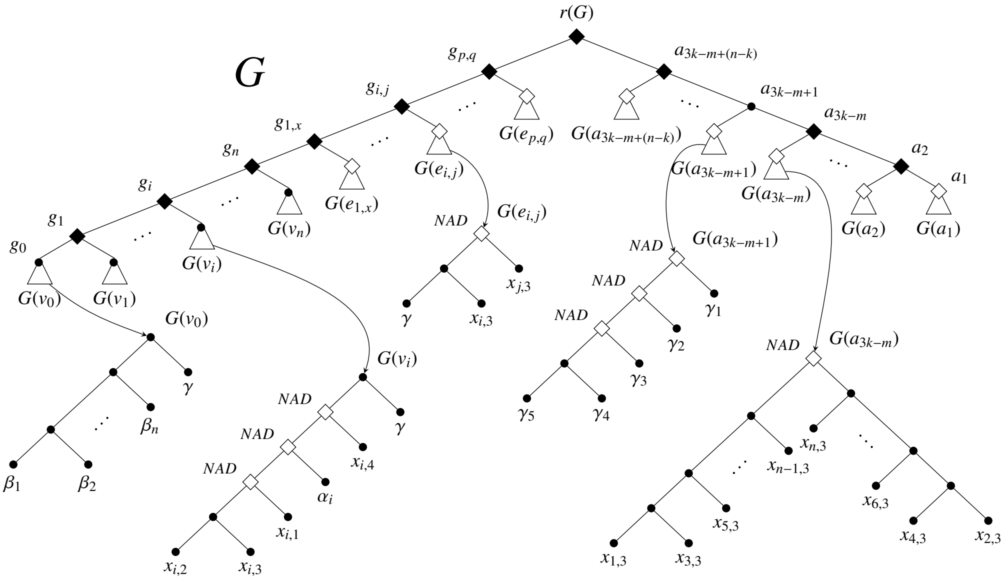

Fig. 3. The gene tree G associated with an instance of *MinVC on Cubic Graphs*. Speciation nodes are represented as circles, NAD nodes as white diamonds, and AD nodes as black diamonds.

The subtree $T(\gamma)$ is a caterpillar on the ordered set $\langle \gamma, \gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5 \rangle$.

The tree G (see Fig. 3) consists of a left path and a right path. Informally, the left path has one tree for each vertex of $\mathcal{G}$ and for each edge of $\mathcal{G}$, and the leaf insertions in these subtrees are related to the cover of the graph; on the other hand, the right path and the subtrees connected to it are built so that the overall number of leaf insertions is exactly $|M|$. More precisely, the left path connects the root $r(G)$

to nodes $g_{p,q}, \dots g_{1,x}, g_n, \dots, g_1, g_0$, (we assume that the edges of $\mathcal{G}$ are ordered so that $\{v_1, v_x\}$ is the first edge and $\{v_p, v_q\}$ is the last edge according to some ordering of the edges of $\mathcal{G}$). Each node g_i, with $1 \leq i \leq n$, is connected to a subtree $G(v_i)$, each node $g_{i,j}$, with $\{v_i, v_j\} \in E$, is connected to a subtree $G(e_{i,j})$.

Subtree $G(v_0)$ is a caterpillar on the ordered set $\langle \beta_1, \dots, \beta_n, \gamma \rangle$. The subtree $G(v_i)$, with $1 \leq i \leq n$, is a caterpillar on the ordered set $\langle x_{i,2}, x_{i,3}, x_{i,1}, \alpha_i, x_{i,4}, \gamma \rangle$.

The subtree $G(e_{i,j})$, with $e_{i,j} = \{v_i, v_j\} \in E$, is a caterpillar on the ordered set $\langle \gamma, x_{i,3}, x_{j,3} \rangle$.

The right path of G connects the root $r(G)$ to nodes $a_{3k-m+(n-k)}, \dots, a_1$. W.l.o.g. we assume that n is even. Each node a_i, with $1 \leq i \leq 3k - m + (n-k)$, is connected to a subtree $G(a_i)$.

For each i with $1 \leq i \leq 3k - m$, $G(a_i)$ is a copy of a tree consisting of two caterpillars: a left caterpillar on the ordered set $\langle x_{1,3}, x_{3,3}, \dots, x_{n-1,3} \rangle$ and a right caterpillar on the ordered set $\langle x_{2,3}, x_{4,3}, \dots, x_{n,3} \rangle$.

For each i with $3k-m+1 \leq i \leq 3k-m+(n-k)$, $G(a_i)$ is a copy of a caterpillar on the ordered set $\langle \gamma_5, \gamma_4, \gamma_3, \gamma_2, \gamma_1 \rangle$.

Finally, we define the multiset M. M contains three occurrences of label $x_{i,3}$, for each i with $1 \leq i \leq n$, and four occurrences of label β_i, for each i with $1 \leq i \leq n$.

We start by stating some properties of trees G, T (see Fig. 2 and Fig. 3).

Remark 1 *Let $\mathcal{G}$ be a cubic graph and let (G, T, M) be the associated instance of LeafIns. Then, there exist three NAD nodes in each $G(v_i)$, with $1 \leq i \leq n$, one NAD node in each $G(e_{i,j})$, with $\{v_i, v_j\} \in E$, one NAD node in each $G(a_i)$, with $1 \leq i \leq 3k-m$, three NAD nodes in each $G(a_i)$, with $3k-m+1 \leq i \leq 3k-m+(n-k)$.*

Each node g_i, with $1 \leq i \leq n$, is an AD node, each node $g_{i,j}$, with $\{v_i, v_j\} \in E$, is an AD node, each node a_i, with $2 \leq i \leq 3k-m$, is an AD node (a_1 is a NAD node). Each node a_i, with $3k-m+2 \leq i \leq 3k-m+(n-k)$, is an AD node, while node a_{3k-m+1} is a speciation node.

The main idea of the reduction is to insert leaves with labels in M so that the NAD nodes in G become AD nodes. Each subtree $G(v_i)$, with $1 \leq i \leq n$, can be made consistent with T essentially in two possible ways (see Lemma 3.1): either by inserting three leaves labeled by $x_{i,3}$ (this corresponds to the case that vertex v_i is not in the vertex cover of $\mathcal{G}$) or by inserting four leaves labeled by β_i (this corresponds to the case that v_i is in the vertex cover of $\mathcal{G}$). The single NAD node of $G(e_{i,j})$ (see Lemma 3.2 and Lemma 3.5) becomes an AD node by inserting a leaf labeled either by $x_{i,3}$ or by $x_{j,3}$ (depending on the fact that v_i or v_j is in the vertex cover of $\mathcal{G}$).

Each subtree $G(a_i)$, with $1 \leq i \leq 3k-m$, is made consistent with T by inserting a leaf labeled by $x_{j,3}$, with $1 \leq j \leq n$, where $x_{j,3}$ is not used to label any inserted leaf of $G(v_j)$ or of $G(e_{j,h})$. Each subtree $G(a_i)$, with $3k-m+1 \leq i \leq 3k-m+(n-k)$, is made consistent with T by inserting four leaves labeled by β_j, with $1 \leq j \leq n$, where β_j is not used to label any inserted leaves of $G(v_j)$.

Now, we are ready to prove the details of the reduction. Next, we give some properties of subtrees $G(v_i)$, and $G(a_j)$.

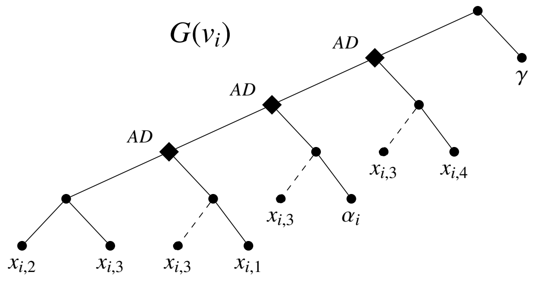

Fig. 4. A subtree $G(v_i)$ made consistent with T by inserting three leaves labeled by $x_{i,3}$.

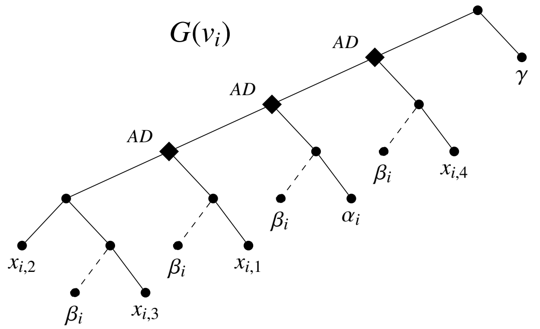

Fig. 5. A subtree $G(v_i)$ made consistent with T by inserting four leaves labeled by β_i.

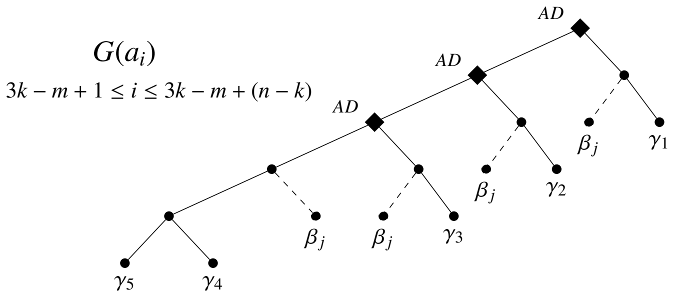

Fig. 6. A subtree $G(a_i)$, with $3k - m + 1 \leq i \leq 3k - m + (n - k)$, made consistent with T by inserting four leaves labeled by β_j.

Remark 2 *Let $\mathcal{G}$ be a cubic graph and let (G, T, M) be the associated instance of LeafIns. Then:*

- *$G(v_i)$, with $1 \leq i \leq n$, can be made consistent with T by inserting three leaves labeled $x_{i,3}$ (see Fig. 4) or by inserting four leaves labeled by β_i (see Fig. 5).*
- *$G(a_i)$, with $1 \leq i \leq 3k - m$, can be made consistent with T by inserting a leaf labeled by $x_{j,3}$, with $1 \leq j \leq n$.*
- *$G(a_i)$, with $3k - m + 1 \leq i \leq 3k - m + (n - k)$, can be made consistent with T by inserting four leaves labeled by β_j, with $1 \leq j \leq n$ (see Fig. 6).*

Moreover, we can prove the following property for subtrees $G(v_i)$, with $1 \leq i \leq n$.

Lemma 3.1 *Given a vertex $v_i \in V$, let $G(v_i)$ be the corresponding subtree of G. Then a subtree $G^*(v_i)$ consistent with T is obtained either by inserting three leaves* *labeled by $x_{i,3}$ or by inserting at least four leaves.*

Proof. By Remark 2, it follows that, starting from $G(v_i)$, it is possible to compute a subtree $G^*(v_i)$ consistent with T by inserting three leaves labeled by $x_{i,3}$. It is easy to see that if a subtree $G^*(v_i)$ consistent with T is computed by inserting only leaves labeled by y, with $y \in M$ and $y \neq x_{i,3}$, then $G^*(v_i)$ can be computed by inserting four leaves, since $y \notin \Lambda(G(v_i))$.

Finally, assume that a subtree $G^*(v_i)$ is obtained by inserting in $G(v_i)$ at least one leaf (and at most two leaves) labeled by $x_{i,3}$, and at least one leaf labeled by y, with $y \in M$ and $y \neq x_{i,3}$. Now, consider the three NAD nodes of $G(v_i)$ (see Remark 1). Since at most two inserted leaves are labeled by $x_{i,3}$, there must exist a node of $G^*(v_i)$ whose left and right subtrees both contain leaves labeled by y. For each other NAD node of $G(v_i)$, at least one leaf must be inserted (labeled by $x_{i,3}$ or by a label different from $x_{i,3}$) and the lemma follows. □

Next, we prove a property of subtrees $G(e_{i,j})$.

Lemma 3.2 *Consider an edge $\{v_i, v_j\} \in E$ and the corresponding subtree $G(e_{i,j})$ in G. Then a subtree $G^*(e_{i,j})$ consistent with T is obtained by inserting in $G(e_{i,j})$ either one leaf with label in $\{x_{i,3}, x_{j,3}\}$, or at least two leaves.*

Proof. Notice that $G(e_{i,j})$ contains one NAD node (see Remark 1) that can be made an AD node by inserting a leaf labeled by $x_{i,3}$ ($x_{j,3}$, respectively) as a sibling of the leaf of $G(e_{i,j})$ labeled by $x_{j,3}$ ($x_{i,3}$, respectively). Finally, since $\Lambda(G(e_{i,j})) \cap M = \{x_{i,3}, x_{j,3}\}$, if no leaf with label in $\{x_{i,3}, x_{j,3}\}$ is inserted in $G(e_{i,j})$, then a subtree $G^*(e_{i,j})$ consistent with T is obtained by inserting in $G(e_{i,j})$ at least two leaves. □

Next, we prove a bound on the number of leaves that has to be inserted by a solution of LeafIns over instance (G, T, M) in the subtrees $G(v_i)$, with $1 \leq i \leq n$, and $G(e_{i,j})$, with $\{v_i, v_j\} \in E$.

Lemma 3.3 *Given a cubic graph $\mathcal{G}$ and the corresponding instance (G, T, M) of LeafIns, a solution G^* of LeafIns over instance (G, T, M) inserts at most $3(n-k) + 4k + m$ leaves in the subtrees $G(v_i)$, with $v_i \in V$, and $G(e_{i,j})$, with $\{v_i, v_j\} \in E$.*

Proof. Let G^* be a gene tree consistent with T obtained by inserting leaves labeled by M. First, notice that each subtree $G(a_i)$, with $1 \leq i \leq 3k - m + (n-k)$, can be made consistent by inserting at least $4(n-k) + 3k - m$ leaves. Indeed each subtree $G(a_i)$, with $1 \leq i \leq 3k - m$, contains one NAD node and requires at least one leaf insertion, while each subtree $G(a_i)$, with $3k - m + 1 \leq i \leq 3k - m + (n-k)$, contains four NAD nodes and requires at least four leaf insertions. Since $|M| = 7n$, the subtrees $G(v_i)$, with $1 \leq i \leq n$, and $G(e_{i,j})$, with $\{v_i, v_j\} \in E$, must be made consistent with T by inserting at most $7n - (3k - m + 4(n-k)) = 3n + m + k = 3(n-k) + 4k + m$ leaves. □

Lemma 3.4 *Given a cubic graph $\mathcal{G}$ and the corresponding instance (G, T, M) of LeafIns, a solution G^* of LeafIns over instance (G, T, M) contains at least $(n - k')$ subtrees $G^*(v_i)$, with $1 \leq i \leq n$ and $k' \leq k$, where at most three leaves are inserted,* *and at most $(k - k')$ subtrees $G^*(e_{i,j})$, with $\{v_i, v_j\} \in E$, where at least two leaves are inserted.*

Proof. Consider a solution G^* of LeafIns over instance (G, T, M). Notice that $(n - k') \geq (n - k)$ else, by Lemma 3.3, G^* cannot be a solution of LeafIns over instance (G, T, M).

Assume that there exist m' subtrees $G^*(e_{i,j})$ in G^*, with $\{v_i, v_j\} \in E$, where at least two leaves are inserted. It follows that the number of leaves that have been inserted in subtrees $G^*(v_i)$, with $1 \leq i \leq n$, and $G^*(e_{i,j})$, with $\{v_i, v_j\} \in E$, are at least $3(n - k') + 4k' + m + m'$. Since by Lemma 3.3 at most $3(n - k) + 4k + m$ leaves are inserted in subtrees $G(v_i)$, with $1 \leq i \leq n$, and in subtrees $G(e_{i,j})$, with $\{v_i, v_j\} \in E$, it follows that $3(n - k') + 4k' + m + m' \leq 3(n - k) + 4k + m$, which implies $3n + k' + m + m' \leq 3n + k + m$, hence $m' \leq (k - k')$. □

In the next lemma, we prove that in a solution of LeafIns over instance (G, T, M) each subtree $G(e_{i,j})$, with $\{v_i, v_j\} \in E$, is essentially modified by inserting a single leaf labeled either by $x_{i,3}$ or by $x_{j,3}$.

Lemma 3.5 *Given a cubic graph $\mathcal{G}$ and the corresponding instance (G, T, M) of LeafIns, consider a solution G^* of LeafIns over instance (G, T, M). Then, there exists a solution G^+ of LeafIns over instance (G, T, M) such that, for each $\{v_i, v_j\} \in E$, a single leaf labeled either by $x_{i,3}$ or by $x_{j,3}$ has been inserted in $G^+(e_{i,j})$.*

Proof. Let G^* be a solution of LeafIns over instance (G, T, M).

Consider the k' subtrees $G^*(v_i)$, with $1 \leq i \leq n$, where exactly three leaves are inserted. Assume that there exist two subtrees $G^*(v_i)$ and $G^*(v_j)$, with $1 \leq i < j \leq n$ and $\{v_i, v_j\} \in E$, where three leaves are inserted. By construction and by Lemma 3.2, it follows that subtree $G^*(e_{i,j})$ is obtained by inserting at least two leaves. We compute a solution G^+ of LeafIns over instance (G, T, M) such that if two subtrees $G^+(v_i)$ and $G^+(v_j)$ are obtained by inserting exactly three leaves, then $\{v_i, v_j\} \notin E$.

Define now the following three sets of subtrees, $\mathcal{F}_{\mathcal{I}}$, $\mathcal{F}_{\mathcal{C}}$, $\mathcal{F}_{\mathcal{E}}$, such that $\mathcal{F}_{\mathcal{I}}$ contains all the subtrees $G^*(v_i)$ where three leaves have been inserted, $\mathcal{F}_{\mathcal{C}}$ is initially empty and $\mathcal{F}_{\mathcal{E}}$ contains the subtrees $G^*(e_{i,j})$ such that $G^*(v_i)$ and $G^*(v_j)$ are both in $\mathcal{F}_{\mathcal{I}}$.

Starting from $\mathcal{F}_{\mathcal{I}}$, $\mathcal{F}_{\mathcal{C}}$, $\mathcal{F}_{\mathcal{E}}$, we compute three sets $\mathcal{F}_{\mathcal{I},\mathcal{Q}}$, $\mathcal{F}_{\mathcal{C},\mathcal{Q}}$, $\mathcal{F}_{\mathcal{E},\mathcal{Q}}$ with an iterative procedure that, while $\mathcal{F}_{\mathcal{E}}$ is not empty, picks a subtree $G^*(e_{i,j})$ in $\mathcal{F}_{\mathcal{E}}$, puts one of the trees $G^*(v_i)$ and $G^*(v_j)$ in $\mathcal{F}_{\mathcal{C}}$ and removes it from $\mathcal{F}_{\mathcal{I}}$. Then, the procedure updates $\mathcal{F}_{\mathcal{E}}$, that is, it removes those subtrees $G^*(e_{i,j})$ such that at most one of $G^*(v_i)$ and $G^*(v_j)$ belongs to $\mathcal{F}_{\mathcal{I}}$.

Consider now the sets $\mathcal{F}_{\mathcal{I},\mathcal{Q}}$, $\mathcal{F}_{\mathcal{C},\mathcal{Q}}$, $\mathcal{F}_{\mathcal{E},\mathcal{Q}}$ obtained from $\mathcal{F}_{\mathcal{I}}$, $\mathcal{F}_{\mathcal{C}}$, $\mathcal{F}_{\mathcal{E}}$ when the procedure terminates. Notice that by construction $\mathcal{F}_{\mathcal{E},\mathcal{Q}}$ is empty. Since at each step at least one tree is removed from $\mathcal{F}_{\mathcal{E}}$ and exactly one subtree is added to $\mathcal{F}_{\mathcal{C}}$ (and removed from $\mathcal{F}_{\mathcal{I}}$), the following claim holds.

Claim 3.6 $|\mathcal{F}_{\mathcal{C},\mathcal{Q}}| \leq |\mathcal{F}_{\mathcal{E}}|$ *and* $|\mathcal{F}_{\mathcal{I}}| - |\mathcal{F}_{\mathcal{I},\mathcal{Q}}| \leq |\mathcal{F}_{\mathcal{E}}|$.

Since by Lemma 3.4 the subtrees in $|\mathcal{F}_{\mathcal{E}}|$ are at most $k - k'$, it follows by Claim 3.6 that $|\mathcal{F}_{\mathcal{I}}| - |\mathcal{F}_{\mathcal{I},\mathcal{Q}}| \leq |\mathcal{F}_{\mathcal{E}}| \leq k - k'$. This implies that $|\mathcal{F}_{\mathcal{I},\mathcal{Q}}| \geq (n - k') - (k - k') = n - k$.

Without loss of generality we assume that $|\mathcal{F}_{\mathcal{I},\mathcal{Q}}| = n - k$ (otherwise some subtrees can be deleted from $\mathcal{F}_{\mathcal{I},\mathcal{Q}}$).

Now, we are able to define the solution G^+ of LeafIns as follows:

- for each subtree $G^*(v_i)$ in $\mathcal{F}_{\mathcal{I},\mathcal{Q}}$, compute $G^+(v_i)$ by inserting three leaves labeled by $x_{i,3}$, so that $G^+(v_i)$ is consistent with T (see Remark 2);
- for every other subtree $G^*(v_i)$, compute $G^+(v_i)$ by inserting four leaves labeled by β_i, so that $G^+(v_i)$ is consistent with T (see Remark 2);
- for each subtree $G(e_{i,j})$, compute $G^+(e_{i,j})$ by inserting one leaf labeled either by $x_{i,3}$, if $G^*(v_j)$ is in $\mathcal{F}_{\mathcal{I},\mathcal{Q}}$, or by $x_{j,3}$, if $G^*(v_i)$ is in $\mathcal{F}_{\mathcal{I},\mathcal{Q}}$, so that $G^+(e_{i,j})$ is consistent with T (see Lemma 3.2). Notice that by construction at most one of $G^*(v_i)$, $G^*(v_i)$ belongs to $\mathcal{F}_{\mathcal{I},\mathcal{Q}}$;
- for each subtree $G(a_h)$, with $1 \leq h \leq 3k-m$, insert one leaf $x_{i,3}$, with $G^*(v_i)$ not in $\mathcal{F}_{\mathcal{I},\mathcal{Q}}$, and such that one occurrence of $x_{i,3}$ has not been associated with an inserted leaf in a subtree $G^+(e_{i,j})$ or a different subtree $G(a_q)$, with $1 \leq q \leq 3k-m$, so that the resulting subtree $G^+(a_h)$ is consistent with T (see Remark 2). This is possible since $3(n-k)+m$ labels $x_{i,3}$ are used for leaf insertions in subtrees in $G^+(v_i)$ and $G^+(e_{i,j})$, while M contains $3n$ labels $x_{i,3}$, for some i with $1 \leq i \leq n$; hence $3n-(3n-3k+m)$ labels can be used for trees $G(a_h)$, with $1 \leq h \leq 3k-m$, and since in a cubic graph $m = \frac{3n}{2}$, it follows that there exist $2m-2m+3k-m = 3k-m$ labels $x_{i,3}$, each one used for a leaf insertion in subtrees $G^+(a_h)$, with $1 \leq h \leq 3k-m$;
- for each of subtree $G(a_h)$, with $3k-m+1 \leq h \leq 3k-m+(n-k)$, insert four leaves β_i, for some $G^*(v_i)$ in $\mathcal{F}_{\mathcal{I},\mathcal{Q}}$, and not already associated with inserted leaves in some subtree $G(a_p)$, with $3k-m+1 \leq p \leq 3k-m+(n-k)$, so that the resulting subtree $G^+(a_h)$ is consistent with T (see Remark 2). This is possible since $|\mathcal{F}_{\mathcal{I},\mathcal{Q}}| = n-k$, hence there exist four occurrences of $(n-k)$ labels β_i, used for leaf insertions in subtrees $G^+(a_j)$, with $3k-m \leq j \leq 3k-m+(n-k)$.

Since G^+ is consistent with T, the lemma holds. □

Now, we are ready to present the main result of the reduction.

Lemma 3.7 *Let $\mathcal{G}$ be a cubic graph and let (G, T, M) be the corresponding instance of LeafIns. There exists a solution of* MinVC on Cubic Graphs *of size k if and only if there exists a solution G^* of LeafIns over instance (G, T, M).*

Proof. (⇒) Consider a vertex cover V' of $\mathcal{G}$ consisting of k vertices (some vertex may be added to V' if $|V'| < k$). Then, we define a gene tree G^* obtained from G by inserting the leaves in M as follows:

- for each $v_i \in V'$, insert four leaves labeled by β_i in $G(v_i)$, so that the resulting subtree $G^*(v_i)$ is consistent with T (see Remark 2);
- for each $v_i \in V \setminus V'$, insert three leaves labeled by $x_{i,3}$ in $G(v_i)$, so that the resulting subtree $G^*(v_i)$ is consistent with T (see Remark 2);
- for each $\{v_i, v_j\} \in E$, insert one leaf labeled by $x_{i,3}$ in $G(e_{i,j})$, with $v_i \in V'$, so that the resulting subtree $G^*(e_{i,j})$ is consistent with T (see Lemma 3.2);

- for each of subtree $G(a_i)$, with $1 \leq i \leq 3k - m$, insert one leaf labeled by $x_{h,3}$, with $v_h \in V'$, not associated with a leaf already inserted in some $G(e_{h,j})$ or in a subtree $G(a_p)$, with $1 \leq p \leq 3k - m$, so that the resulting subtree $G^*(a_i)$ is consistent with T (see Remark 2);
- for each of subtree $G(a_i)$, with $3k - m + 1 \leq i \leq 3k - m + (n - k)$, insert four leaves labeled by β_h, for some $v_h \in V'$, not associated with a leaf already inserted in some subtree $G(a_p)$, with $3k - m + 1 \leq p \leq 3k - m + (n - k)$, so that the resulting subtree $G^*(a_i)$ is consistent with T (see Remark 2).

Then each NAD node of G becomes an AD node (see Remark 1) in G^*, hence the constructed gene tree G^* is consistent with T, thus it is a solution of LeafIns over instance (G, T, M).

($\Leftarrow$) Let G^* be a solution of LeafIns over instance (G, T, M). By Lemma 3.3, the subtrees $G(v_i)$, with $1 \leq i \leq n$, and $G(e_{i,j})$, with $\{v_i, v_j\} \in E$, must be made consistent with T by inserting at most $7n - (3k - m + 4(n - k)) = 3(n - k) + 4k + m$ leaves.

Now, given $v_i, v_j \in V$, with $\{v_i, v_j\} \in E$, consider the subtrees $G^*(v_i)$ and $G^*(v_j)$ of G^* consistent with T. Since there exist three occurrences of label $x_{i,3}$ in M, it follows by Lemma 3.5 and by Lemma 3.1 that at most one of $G^*(v_i)$ and $G^*(v_j)$ can be made consistent with T by inserting exactly three leaves.

By Lemma 3.4, there exist at least $(n - k') \geq (n - k)$ trees $G^*(v_i)$ (we assume exactly $n - k$) that are made consistent by inserting three leaves labeled by $x_{i,3}$. By Lemma 3.5 each subtree $G^*(e_{i,j})$, with $\{v_i, v_j\} \in E$, is obtained by inserting a leaf labeled by $x_{i,3}$ or by $x_{j,3}$. Moreover, we can assume that k subtrees $G^*(v_i)$, with $v_i \in V$, consistent with T, are obtained by inserting four leaves labeled by β_i, each subtree $G^*(a_i)$, with $1 \leq i \leq 3k - m$, is obtained by inserting a leaf labeled by $x_{j,3}$ and each subtree $G^*(a_i)$, with $3k - m + 1 \leq i \leq 3k - m + (n - k)$, is obtained by inserting four leaves labeled by β_j.

It follows from Lemma 3.1 and Lemma 3.5 that the set

$$V' = \{v_i : G^*(v_i) \text{ has four leaves labeled by } \beta_i\ \}$$

is a vertex cover of size k, since $|V'| = k$ and for each $\{v_i, v_j\} \in E$ at least one of v_i, v_j is in V'. □

From Lemma 3.7, the following result holds.

Theorem 3.8 *LeafIns is NP-complete.*

Proof. LeafIns is in NP, since given a gene tree G^* obtained by inserting leaves with labels in M, we can check in polynomial time that it is consistent with T.

The NP-hardness of LeafIns follows from Lemma 3.7 and from the NP-hardness of *MinVC on Cubic Graphs* [1]. □

From the NP-completeness of LeafIns follows the NP-hardness of MinLeafIns.

Corollary 3.9 *MinLeafIns is NP-hard.*

Proof. The result follows from Lemma 3.8, since computing if there exists a gene tree consistent with T and obtained by inserting at most $|M|$ leaf insertions is NP-hard. □

4 A 2-approximation Algorithm for MinLeafIns

In this section, we show that MinLeafIns can be approximated in polynomial time within factor 2.

Given an instance (G, T, M) of MinLeafIns, denote by $\sharp_{NAD}$ the number of NAD nodes of G w.r.t. T. The approximation algorithm is based on the following bound.

Lemma 4.1 *Let (G, T, M) be an instance of MinLeafIns. Then, a gene tree G^* compatible with T is obtained by inserting at least* $\max(|M|, \sharp_{NAD})$ *leaves.*

Proof. First, we show that at least $\sharp_{NAD}$ leaf insertions are required to make G consistent with T.

We assume that NAD nodes are sequentially transformed into AD nodes starting from the leaves of the tree, and G is modified accordingly. Moreover, each NAD node x is transformed into an AD node in two possible ways: with a single leaf insertion or with at least two insertions of leaves having the same label.

Now, consider a NAD node x in G and an ancestor y of x which is a NAD node, such that w.l.o.g. x is in $G[y_r]$. At least one leaf must be inserted in $G[x]$ in order to make x an AD node. Moreover, notice that if a single leaf labeled by l is inserted in $G[x]$, then no other ancestor of x (hence no other node of G) can be made an AD node (from a NAD node), due to this insertion. Indeed, since the insertion of the leaf labeled l can make x an AD node, it follows that $l \in \Lambda(G[x])$, hence $l \in \Lambda(G[y_r])$. As a consequence, the insertion of a single leaf labeled by l does not change $\Lambda(G[y_r])$.

Assume that x becomes an AD node due to the insertion of (at least) two leaves having the same label l. Hence, there exists at least one leaf inserted in $G[x_r]$ and at least one leaf inserted in $G[x_l]$. Notice that at most one ancestor of x in G can become an AD node (from a NAD node). Indeed, let y be the farthest NAD node from the root of G and ancestor of x, that becomes an AD node due to the insertion of the leaves labeled by l. Then consider an ancestor z of x and y, such that x and y belong w.l.o.g. to $G[z_r]$. Since y becomes an AD node by the insertion of leaves labeled by l in $G[x]$, it follows that $l \in G[y] \subseteq G[z_r]$. Hence the insertion of leaves labeled by l does not change $\Lambda(G[z_r])$ and the mapping of nodes z and x_r is not changed by the insertion of two leaves labeled by l. It follows that the insertion of the leaves labeled l makes at most two NAD nodes of G (namely x and y) AD nodes.

We can conclude that at least $\sharp_{NAD}$ leaves have to be inserted in G.

Since at least $|M|$ leaves must be inserted in a solution of MinLeafIns over instance (G, T, M), the lemma follows. □

Now, we describe a polynomial time algorithm that requires at most $2\max(|M|, \sharp_{NAD})$ leaf insertions. Consider the sequence s_{NAD} of NAD nodes in G, visited from the leaves to the root.

The algorithm greedily picks the first node x of s_{NAD} and makes it an AD node as follows:

Step 1 If there exists an occurrence of label l in M such that l is in exactly one of $\Lambda(G[x_l])$, $\Lambda(G[x_r])$ (assume w.l.o.g. in $\Lambda(G[x_l])$), then a leaf labeled by l is inserted in $G[x_r]$ (from Lemma 2.2 it is possible to insert a leaf labeled by l so that the new inserted node in G is not a NAD node); an occurrence of l is removed from M and x is removed from s_{NAD}.

Step 2 Else, insert a leaf associated with a label l such that l is in $\Lambda(G[x])$ (again from Lemma 2.2 it is possible to insert a leaf labeled by l so that the new inserted node in G is not a NAD node); x is removed from s_{NAD}.

After the execution of Step 1 and 2, s_{NAD} is empty, while M may not be empty. Let M_e be such a multiset of labels left after the execution of Step 1 and Step 2. Then, the algorithm removes one occurrence of a leaf labeled by l in M_e, and inserts a leaf labeled by l as a sibling of a leaf of G that has label l. The node introduced by the leaf insertion is an AD node, and the mapping of other nodes of G is not influenced by this leaf insertion.

Next, we prove that the algorithm gives a 2-approximation.

Theorem 4.2 *MinLeafIns can be approximated in polynomial time within factor* 2.

Proof. Consider the case that $M_e = \emptyset$ after the iteration of Step 1 and Step 2. By construction, Step 1 and Step 2 insert at most one leaf for each NAD node, hence at most $\sharp_{NAD}$ leaves. By Lemma 4.1, the optimal solution inserts at least $\sharp_{NAD}$ leaves, hence in this case the result follows.

Consider the case that $M_e \neq \emptyset$. By construction, Step 1 and Step 2 insert at most one leaf for each NAD node. Since $|M_e| \leq |M|$, the algorithm inserts at most $|M| + \sharp_{NAD}$ leaves. By Lemma 4.1 an optimal solution of MinLeafIns inserts at least $\max(|M|, \sharp_{NAD})$ leaves, thus also in this case the algorithm gives a 2-approximation. □

5 Conclusion

In this paper, we have considered a leaf edit operation, leaf insertion, for the problem of correcting a gene tree. First, we have considered a decision variant of the problem, called LeafIns, that, given a gene tree G, a species tree T and a multiset of labels M, asks if there exists a gene tree G^* consistent with T, obtained by inserting leaves having labels in M. We have shown that LeafIns is NP-complete. Then, we have considered an optimization variant of the problem, called MinLeafIns, that, given a gene tree G, a species tree T and a multiset of labels M, asks if there exists a gene tree G^* consistent with T, obtained by inserting leaves labeled by a multiset set M' of minimum cardinality such that $M' \supseteq M$. Notice that MinLeafIns is NP-hard from the hardness result of LeafIns. We have given a polynomial time 2-approximation algorithm for MinLeafIns.

A future direction of research is the investigation of the parameterized complexity of the two problems that we have considered in this paper, since the number of corrections is usually small and thus can be considered as a parameter. Moreover, it would be interesting to study a general variant of the gene tree correction problem, following the approaches in [29,13,5], where different leaf edit operations (leaf insertions, leaf deletions, leaf modifications) are considered.

Acknowledgement

The authors acknowledge the support of the MIUR PRIN 2010-2011 grant 2010LYA9RH (Automi e Linguaggi Formali: Aspetti Matematici e Applicativi)

References

[1] Alimonti, P. and V. Kann, *Some APX-completeness results for cubic graphs*, Theoretical Computer Science **237** (2000), pp. 123–134.

[2] Arvestad, L., A.-C. Berglung, J. Lagergren and B. Sennblad, *Gene tree reconstruction and orthology analysis based on an integrated model for duplications and sequence evolution.*, in: D. Gusfield, editor, *RECOMB 2004* (2004), pp. 326–335.

[3] Bansal, M. S., E. J. Alm and M. Kellis, *Efficient algorithms for the reconciliation problem with gene duplication, horizontal transfer and loss*, Bioinformatics **28** (2012), pp. 283–291.

[4] Bansal, M. S. and R. Shamir, *A note on the fixed parameter tractability of the gene-duplication problem*, IEEE/ACM Trans. Comput. Biology Bioinform. **8** (2011), pp. 848–850.

[5] Beretta, S., M. Castelli and R. Dondi, *Correcting gene tree by removal and modification: Tractability and approximability*, Journal of Discrete Algorithms (to appear).

[6] Blin, G., P. Bonizzoni, R. Dondi, R. Rizzi and F. Sikora, *Complexity insights of the minimum duplication problem*, Theor. Comput. Sci. **530** (2014), pp. 66–79.

[7] Bonizzoni, P., G. Della Vedova and R. Dondi, *Reconciling a gene tree to a species tree under the duplication cost model.*, Theoretical Computer Science **347** (2005), pp. 36–53.

[8] Chang, W. and O. Eulenstein, *Reconciling gene trees with apparent polytomies*, in: D. Chen and D. T. Lee, editors, *COCOON 2006*, LNCS **4112**, Heidelberg, 2006, pp. 235–244.

[9] Chauve, C., J.-P. Doyon and N. El-Mabrouk., *Gene family evolution by duplication, speciation and loss*, J. Comput. Biol. **15** (2008), pp. 1043-1062.

[10] Chauve, C. and N. El-Mabrouk, *New perspectives on gene family evolution: losses in reconciliation and a link with supertrees*, in: S. Batzoglou, editor, *RECOMB 2009*, LNCS **5541** (2009), pp. 46-58.

[11] Chen, K., D. Durand and M. Farach-Colton, *Notung: Dating gene duplications using gene family trees*, Journal of Computational Biology **7** (2000), pp. 429–447.

[12] DasGupta, B., S. Ferrarini, U. Gopalakrishnan and N. R. Paryani, *Inapproximability results for the lateral gene transfer problem*, J. Comb. Optim. **11** (2006), pp. 387–405.

[13] Dondi, R., N. El-Mabrouk and K. M. Swenson, *Gene tree correction for reconciliation and species tree inference: Complexity and algorithms*, Journal of Discrete Algorithms **25** (2014), pp. 51–65.

[14] Doyon, J.-P., C. Scornavacca, K. Gorbunov, G. Szolloso, V. Ranwez and V. Berry, *An effi. algo. for gene/species trees parsim. reconc. with losses, dup. and transf.*, J. Comp. Biol. **6398** (2010), pp. 93- 108.

[15] Durand, D., B. Haldórsson and B. Vernot, *A hybrid micro-macroevolutionary approach to gene tree reconstruction*, Journal of Computational Biology **13** (2006), pp. 320–335.

[16] Eichler, E. and D. Sankoff, *Structural dynamics of eukaryotic chromosome evolution*, Science **301** (2003), pp. 793–797.

[17] Górecki, P. and O. Eulenstein, *Algorithms: simultaneous error-correction and rooting for gene tree reconciliation and the gene duplication problem*, BMC Bioinformatics **13** (2012), p. S14.

[18] Gorecki, P. and J. Tiuryn., *DLS-trees: a model of evolutionary scenarios.*, Theoretical Computer Science **359** (2006), pp. 378–399.

[19] Hahn, M., *Bias in phylogenetic tree reconciliation methods: implications for vertebrate genome evolution*, Genome Biology **8** (2007).

[20] Kordi, M. and M. S. Bansal, *On the complexity of duplication-transfer-loss reconciliation with non-binary gene trees*, in: R. Harrison, Y. Li and I. I. Mandoiu, editors, *Bioinformatics Research and Applications - 11th International Symposium, ISBRA 2015, Norfolk, VA, USA, June 7-10, 2015 Proceedings*, Lecture Notes in Computer Science **9096** (2015), pp. 187–198.

[21] Lafond, M., C. Chauve, R. Dondi and N. El-Mabrouk, *Polytomy refinement for the correction of dubious duplications in gene trees*, Bioinformatics **30** (2014), pp. 519–526.

[22] Ma, B., M. Li and L. Zhang, *From gene trees to species trees*, SIAM J. on Comput. **30** (2000), pp. 729–752.

[23] Ohno, S., "Evolution by gene duplication," Springer, Berlin, 1970.

[24] Page, R., *Maps between trees and cladistic analysis of historical associations among genes, organisms, and areas*, Systematic Biology **43** (1994), pp. 58–77.

[25] Page., R., *Genetree: comparing gene and species phylogenies using reconciled trees.*, Bioinformatics **14** (1998), pp. 819–820.

[26] Page, R. and M. Charleston, *Reconciled trees and incongruent gene and species trees*, DIMACS Series in Discrete Mathematics and Theoretical Computer Science **37** (1997), pp. 57–70.

[27] Page, R. and J. Cotton, *Vertebrate phylogenomics: reconciled trees and gene duplications*, in: *Pacific Symposium on Biocomputing*, 2002, pp. 536–547.

[28] Sanderson, M. and M. McMahon, *Inferring angiosperm phylogeny from EST data with widespread gene duplication.*, BMC Evolutionary Biology **7** (2007), p. S3.

[29] Swenson, K. M., A. Doroftei and N. El-Mabrouk, *Gene tree correction for reconciliation and species tree inference*, Algorithms for Molecular Biology **7** (2012), p. 31.

[30] Tofigh, A., M. Hallett and J. Lagergren, *Simultaneous identification of duplications and lateral gene transfers*, IEEE/ACM Trans. Comput. Biol. Bioinform. **8** (2011), pp. 517-535.

[31] Vernot, B., M. Stolzer, A. Goldman and D. Durand, *Reconciliation with non-binary species trees*, Journal of Computational Biology **15** (2008), pp. 981–1006.