Complexity Insights of the Minimum Duplication Problem

Guillaume Blin, Paola Bonizzoni, Riccardo Dondi, Romeo Rizzi, Florian Sikora

Complexity Insights of the Minimum Duplication Problem

Guillaume Blin[1], Paola Bonizzoni[2], Riccardo Dondi[3], Romeo Rizzi[4], Florian Sikora[1,5]

[1] Université Paris-Est, LIGM - UMR CNRS 8049, France.
{gblin,sikora}@univ-mlv.fr
[2] DISCo, Universitá degli Studi di Milano-Bicocca, - Milano, Italy.
bonizzoni@disco.unimib.it
[3] DSLCSC, Universitá degli Studi di Bergamo, - Bergamo, Italy.
riccardo.dondi@unibg.it
[4] DIMI - Università di Udine - Udine, Italy. Romeo.Rizzi@dimi.uniud.it
[5] Lehrstuhl für Bioinformatik, Friedrich-Schiller-Universität Jena, Germany.

Abstract. The Minimum Duplication problem is a well-known problem in phylogenetics and comparative genomics. Given a set of gene trees, the Minimum Duplication problem asks for a species tree that induces the minimum number of gene duplications in the input gene trees. More recently, a variant of the Minimum Duplication problem, called Minimum Duplication Bipartite, has been introduced in [14], where the goal is to find all *pre-duplications*, that is duplications that precede, in the evolution, the first speciation with respect to a species tree. In this paper, we investigate the complexity of both Minimum Duplication and Minimum Duplication Bipartite problems. First of all, we prove that the Minimum Duplication problem is APX-hard, even when the input consists of five uniquely leaf-labelled gene trees (progressing on the complexity of the problem). Then, we show that the Minimum Duplication Bipartite problem can be solved efficiently by a randomized algorithm when the input gene trees have bounded depth.

1 Introduction

The evolutionary history of the genomes of eukaryotes is the result of a series of evolutionary events, called *speciations*, that produce new species starting from a common ancestor. This evolutionary history has been deeply studied in Computational Biology, and is usually represented using a special type of phylogenetic tree called *species tree* [9]. A *species tree* is a rooted binary tree whose leaves are uniquely labelled by a set Λ representing the extant species, where the common ancestor of the contemporary species is associated with the root of the tree. The internal nodes represent hypothetical ancestral species (and the associated speciations). Speciations are not the only events that influence the evolution. Indeed, there are other events, such as gene duplication, gene loss and lateral gene transfer that, although not leading to new species, are fundamental in evolution. In this paper, we focus on gene duplications which are known to be essential for the evolution of many eukaryotes groups, such as vertebrates, insects and

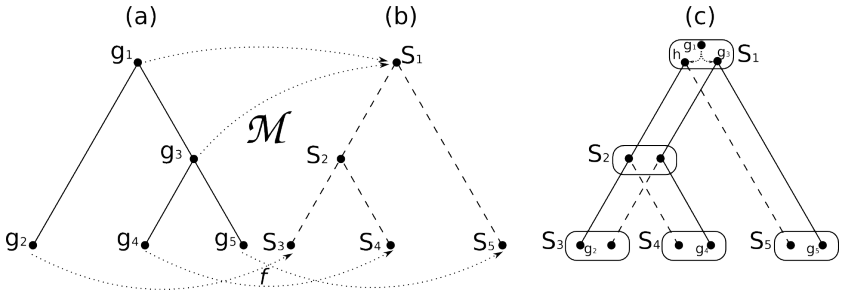

Fig. 1. (a) a gene tree T. (b) a species tree S where $\mathcal{M}$ is the `lca` mapping from T to S; each gene in $\{g_2, g_4, g_5\}$ is mapped by function f in the species that gene belongs to. (c) a reconciled tree for T and S based on the a priori duplication of gene g_1 into genes h and g_3.

plants [8]. Gene duplication can be described as the genomic event that causes a gene inside a genome to be copied, resulting in two copies of the same gene that can evolve independently. Genes of extant species are called *homologous* if they evolved from a common ancestor, through speciations and duplications events [10]. Evolution of homologous genes, with regards to the extant species, is usually represented using another special type of phylogenetic tree called *gene tree*. A *gene tree* is a rooted binary tree whose leaves are (not necessarily uniquely) labelled using elements of the set Λ. Indeed, despite the fact that, biologically speaking, leaves in the gene tree represent genes, for ease, the gene tree is labelled according to the species from which the corresponding gene was sampled. Therefore, leaves similarly labelled represent duplicated genes that evolved independently and appear in a common extant species. As in the species tree, the root and the internal nodes respectively represent the common ancestor and ancestral genes explaining their evolution.

With regards to the set of labels Λ, gene and species trees are said to be *comparable*. Nevertheless, due to complex evolutionary processes such as gene duplication and loss, comparable gene and species trees very often present incompatibilities. A challenging problem is then to reconcile the gene and species trees with hypothetical gene duplications. For example, in Fig. 1, given two comparable gene and species trees inducing incompatibilities, one can infer a reconciled tree based on the *a priori* duplication of gene g_1 into genes h and g_3 (h is a hypothetical ancestor of genes g_2, g_4), which afterwards both speciate according to the topology of the species tree. Based on the principle of parsimony, one is interested in finding the minimum number of gene duplications that can explain all the incompatibilities. This last can be inferred by the so-called *lowest common ancestor mapping* (`lca` mapping), denoted by $\mathcal{M}$ and defined as follows. $\mathcal{M}$ maps any gene of the gene tree to the latest species from which the gene could be sampled. In other words, $\mathcal{M}$ maps each ancestral gene g of the

gene tree to the most recent common ancestor of the extant species from which all the descendant of g were sampled.

For example, in Fig. 1, according to $\mathcal{M}$, g_3 is mapped to S_1 since S_1 is the most recent common ancestor of S_4 and S_5 from which were sampled (represented as a function f) respectively the descendant g_4 and g_5 of g_3. Observe that, considering $\mathcal{M}$, any leaf of the gene tree is mapped to the unique leaf of S similarly labelled (according to Λ). Given $\mathcal{M}$, a gene in the gene tree is a gene duplication if it has a descendant with the same $\mathcal{M}$ mapping. Then, the reconciliation cost is defined as the number of gene duplications in the gene tree induced by the species tree. Computation of this distance has been widely investigated in the context of the Minimum Duplication problem [15,13,11,4], where given a set of gene trees, the objective is to compute a species tree that induces a minimum number of gene duplications.

The Minimum Duplication problem is known to be NP-hard [13]. More recently, the Minimum Duplication problem has been related to the Minimum Triplets Consistency [4]. The complexity of Minimum Triplets Consistency has been deeply studied, and the problem is known to be W[2]-hard [5] and inapproximable within factor $O(\log n)$ [5]. These results coupled with the reduction provided in [4] implies that the Minimum Duplication is NP-hard, W[2]-hard (despite of [15]) and inapproximable within factor $O(\log n)$ even in the specific case of a forest composed of an unbounded number of uniquely leaf-labelled gene trees with three leaves [4]. Therefore, different heuristics and Integer Linear Programs have been developed [2,3,7,6]. Recently, the Minimum Duplication Bipartite problem has been introduced to tackle the Minimum Duplication problem [14]. The Minimum Duplication Bipartite problem corresponds to finding all *pre-duplications*; that is duplications that precede, in the evolution, the first speciation with respect to a species tree. Roughly, this means that only the first level of the species tree is of importance. Indeed, one is interested in knowing if a given species belongs to the subtree of S rooted at the left child of the root or at the right one. Therefore, one can view the species tree as a bipartition (Λ_1, Λ_2) of the set of species Λ. Solving the Minimum Duplication Bipartite problem recursively produces a natural greedy heuristic for the Minimum Duplication problem. The Minimum Duplication Bipartite problem was shown to be 2-approximable [14], but its complexity remains open.

In this contribution, we provide results relying both on the Minimum Duplication problem and the Minimum Duplication Bipartite problem. First of all, we prove that the Minimum Duplication problem is APX-hard, even when the input consists of five uniquely leaf-labeled gene trees (that is for a bounded number of gene trees). Then, we show that the Minimum Duplication Bipartite problem can be solved efficiently by a randomized algorithm when the input gene trees have bounded depth. greedy heuristic for the Minimum Duplication problem. Due to space consideration, we do not provide full details and proofs which are deferred to the full version of the paper.

2 On a tight inapproximability

We present a reduction from MINIMUM VERTEX COVER on cubic graphs (MVCC) to the specific case of the MINIMUM DUPLICATION problem – denoted MIN-5-DUP – where given a set of five uniquely leaf labelled gene trees $\mathcal{F} = \{T_1, T_2, T_3, T_4, T_5\}$, the objective is to compute a species tree S that induces a minimum number of gene duplications (afterwards denoted as $d(\mathcal{F}, S)$). Let $G = (V_G, E_G)$ be a cubic graph (*i.e.* every vertex has degree three), MVCC problem asks for a subset $V'_G \subseteq V_G$, such that for each edge $(v_i, v_j) \in E_G$, at least one of $\{v_i, v_j\}$ belongs to V'_G. In a first step, starting from any cubic graph $G = (V_G, E_G)$, we will construct an associated input $\mathcal{F} = \{T_1, \ldots, T_5\}$ of MIN-5-DUP. Then, we will demonstrate that any species tree S such that $d(\mathcal{F}, S) < q = 6|E_G| + 3|V_G| + 1$ must be *canonical* (defined afterwards). Finally, we will prove that our construction is indeed an L-reduction.

In order to define formally the gene trees, let us first define the central notion of *comb graph*. We will consider a specific subclass of comb graphs corresponding to a binary tree where all the internal nodes lie on a single simple (*i.e.* with no repeated vertices) path referred as the *spine*. For ease, we will nevertheless use the term comb graph in the following to denote those last. Given a sequence $L = \langle l_1, \ldots, l_k \rangle$ of k labels, let $C(L)$ denote the comb graph whose leaves are labelled according to a postorder traversal using L (*i.e.* $l_x \in L$ is the label of the unique leaf of depth x). For example, in Fig. 1, the gene tree (a) corresponds to the comb graph $C(\langle g_2, g_4, g_5 \rangle)$.

Let us now define two operations on trees. Let $T_1 \triangle T_2$ be a tree obtained from two trees T_1 and T_2, by connecting the roots of T_1 and T_2 to a new vertex v which becomes the root of $T_1 \triangle T_2$. Inserting T_2 in the edge e of T_1 will denote the operation that leads to a tree obtained from T_1 and T_2 by replacing the edge $e = (v, v')$ in T_1 by two edges (v, w) and (w, v') and connecting the root of T_2 to the new vertex w.

We are now ready to define the gene trees $T_1, \ldots, T_5$. Roughly, we will associate to each vertex $v \in V_G$, a specific tree T_v and to each edge $e \in E_G$, two trees T^1_e, T^2_e. These trees will be then combined to build the gene trees $T_1, \ldots, T_5$. For ease, let us consider the following order of edges of E_G, $\langle e_1, e_2, \ldots e_{|E_G|} \rangle$ s.t. $\forall e_x = (v_i, v_j)$, $e_y = (v_h, v_k)$ with $x < y$, $i < j$ and $h < k$, either $(i < h)$ or $(i = h$ and $j < k)$. According to this order, we define the following three sequences of labels: $M_1 = \langle m^1_1, m^1_2, \ldots m^1_{|E_G|} \rangle$, $M_2 = \langle m^2_1, m^2_2, \ldots m^2_{|E_G|} \rangle$ and $L = \langle l^1_1, l^2_1, l^1_2, l^2_2, \ldots l^1_{|E_G|}, l^2_{|E_G|} \rangle$. Roughly, any edge e_x is represented by the four labels $\{m^1_x, m^2_x, l^1_x, l^2_x\}$. First of all, for any edge $e_x \in E_G$, let us build the two trees $T^1_{e_x} = C(\langle l^1_x, m^2_x, l^2_x \rangle)$ and $T^2_{e_x} = C(\langle l^2_x, m^2_x, l^1_x \rangle)$. Moreover, for any $v \in V_G$ s.t. v is incident to the edges e_x, e_y and e_z, we build a tree $T_v = \big(C(\langle m^1_x, m^1_y, m^1_z \rangle) \triangle C(\langle l^1_x, l^1_y, l^1_z \rangle)\big) \triangle C(\langle m^2_x, m^2_y, m^2_z \rangle)$ (see Fig. 2).

We will now build the gene trees T_1 to T_5 by starting from a comb graph where subtrees representing vertices and edges will be inserted in. Let T_5 be obtained from $C(\langle f_{|V_G|+1}, f^1_{|V_G|}, \ldots, f^q_{|V_G|}, f_{|V_G|}, \ldots f^1_1, \ldots, f^q_1, f_1 \rangle)$, by inserting in the edge connecting f_1 and its parent the subtree $C(M_1) \triangle C(M_2)$. Regarding

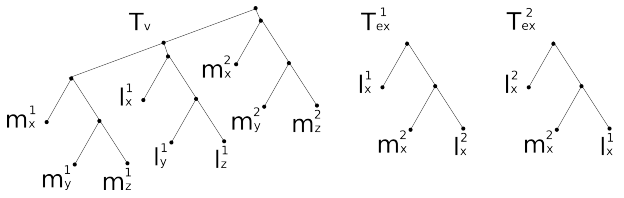

Fig. 2. The trees T_v, $T^1_{e_x}$ and $T^2_{e_x}$ for $v \in V_G$ incident to the edges e_x, e_y and e_z.

the construction of T_1 to T_4, let us assume that we are also provided a 4-coloring $\lambda : V_G \mapsto \{1, 2, 3, 4\}$ of G (for example, by applying the polynomial-time greedy Welsh-Powell algorithm [16]). Let any T_i, $1 \leq i \leq 4$, be first define as a the following comb graph: $T_i = C(\langle f_1, f_2, \ldots, f_{|V_G|+1}, f^1_{|V_G|}, \ldots f^q_{|V_G|}, f^1_{|V_G|-1} \ldots, f^q_1 \rangle)$. We then insert, for each $v_i \in V_G$, the tree T_{v_i} in the edge connecting the parents of f_i and f_{i+1} in the gene tree T_x where $x = \lambda(v_i)$ (see Fig. 3). Moreover, for each $e_x = (v_i, v_j) \in E_G$ (ordered from e_1 to $e_{|E_G|}$), the tree $T^1_{e_x}$ is inserted in the edge connecting the parent of f_i and its other child in the gene tree T_x where $x = min\{1, 2, 3, 4\} \setminus \{\lambda(v_i), \lambda(v_j)\}$ (*i.e.* the gene tree having the smallest index and not containing neither T_{v_i}, nor T_{v_j}). Finally, for each $e_x = (v_i, v_j) \in E_G$, the tree $T^2_{e_x}$ is inserted in the edge connecting the parent of f_i and its other child in the gene tree T_x where $x = max\{1, 2, 3, 4\} \setminus \{\lambda(v_i), \lambda(v_j)\}$ (*i.e.* the gene tree having the biggest index and not containing neither T_{v_i}, nor T_{v_j}). A sketch of this construction is given in Fig. 3.

Due to space constraints, we only provide here a sketch of our proof (full details available in appendix). First of all, we can prove that, by construction, all the gene trees are indeed uniquely leaf-labelled. Then, we can prove that only *canonical* solutions are of interest. Roughly, a canonical solution (*i.e.* a species tree S) is a copy of T_5 where extra leaves of $L = \{l^1_x, l^2_x : \forall e_x \in E_G\}$ are each inserted either in $C(M_1)$ or $C(M_2)$. The insertion of l^1_x, l^2_x in $C(M_1)$ or $C(M_2)$ depends on the fact that the edge $e_x = (v_i, v_i)$ is covered by v_i or v_j.

We can, moreover, prove that, in a canonical solution, (1) each vertex on the path from the root of T_j, with $1 \leq j \leq 4$, to the parent of $f_{|V_G|+1}$ (excluding this last) induces a duplication (that is $5|V_G| + 2|E_G|$ in total), (2) each edge $e_x = (v_i, v_j) \in E_G$ induces a duplication in the root of one of $\{T^1_{e_x}, T^2_{e_x}\}$ and a duplication in the root of either T_{v_i} or T_{v_j}. One can then easily see that the minimum number of duplications is then related to the minimum cover size. Hence the following lemma holds.

Lemma 1. *Let $G = (V_G, E_G)$ be an instance of* MVCC *and let $\mathcal{F} = \{T_1, \ldots, T_5\}$ be the corresponding instance of* Min-5-Dup*. Then, starting from a cover V'_G of G, we can compute in polynomial time a solution S of* Min-5-Dup *for $\mathcal{F}$ s.t. $d(\mathcal{F}, S) \leq 5|V_G| + 3|E_G| + |V'_G|$; starting from a solution S of* Min-5-Dup *for $\mathcal{F}$ s.t. $d(\mathcal{F}, S) \leq 5|V_G| + 3|E_G| + p$, we can compute in polynomial time a cover of G of size at most p.*

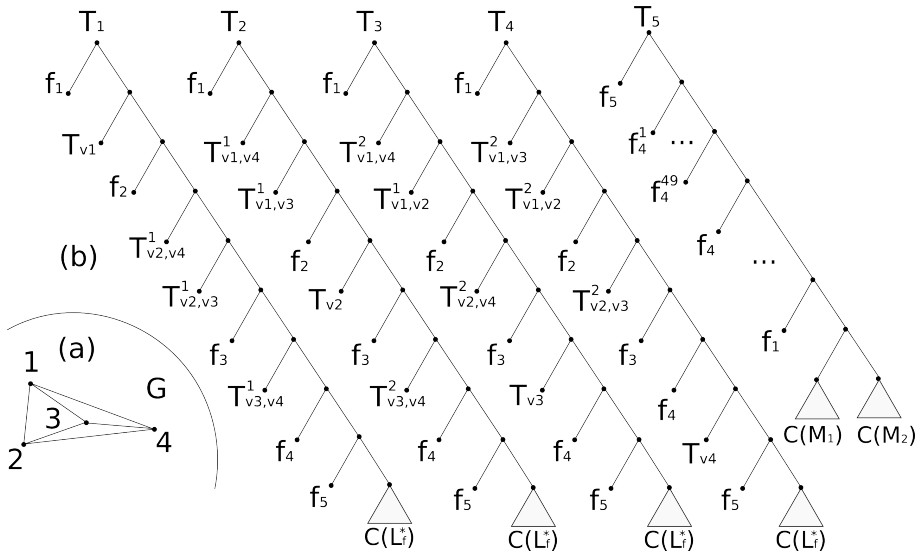

Fig. 3. Gene trees T_1 to T_5 obtained from the cubic graph G where $L_f^* = \langle f^1_{|V_G|}, \ldots, f^q_{|V_G|}, \ldots, f^1_1, \ldots, f^q_1 \rangle$ and $\forall 1 \leq i \leq 4, \lambda(v_i) = i$.

Lemma 1 concludes the reduction. Since MVCC is APX-hard [1], provided our L-reduction, we can conclude that Min-5-Dup is also APX-hard.

Theorem 1. *The* Minimum Duplication *problem is* APX*-hard, even when the input consists of five uniquely leaf-labelled gene trees*

3 A randomized approach

In this section, we investigate the complexity of the Minimum Duplication Bipartite problem and show that it can be solved efficiently by a randomized algorithm when the input gene trees have bounded depth. A randomized algorithm can be seen simply as an algorithm that is allowed to do some random decisions as it processes the input. Whereas defining a randomized algorithm is quite easy, the performance analysis of this last is more complicated. Indeed, first, one has to compute the probability of success of the randomized algorithm (*i.e.* probability to end up with an optimal solution). Then, one can amplify the probability of success simply by repeatedly running the algorithm, with independent random choices, and taking the best solution found. If one, moreover, prove that the overall running time required to get a high probability of success is polynomial in the size of the input, then it implies that the problem is randomized polynomial (in RP-class). For further details on randomized algorithms, the reader should consider the book of Kleinberg and Tardos [12].

In order to prove that the Minimum Duplication Bipartite problem is randomized polynomial, we first provide a randomized algorithm for a variant of the Minimum Cut problem, called Minimum Cut in Colored Graph. Then, we will prove that the Minimum Duplication Bipartite problem can be translated into a Minimum Cut in Colored Hypergraph problem that can be solved efficiently applying our randomized algorithm on hypergraphs with bounded hyperedges degree. It is of importance to note that, as far as we know, this is the first attempt of solving by randomization the minimum cut in colored hypergraph. Providing a randomized algorithm for general hypergaphs with unbounded hyperedges degree is still open.

Let us first introduce the Minimum Cut in Colored Graph problem: Given a set of colors $\mathcal{C}$ and an undirected colored graph $G = (V, E)$ where any edge is colored with a color from $\mathcal{C}$, find a minimal colored cut of G – that is a partition of V into two non-empty sets A and B such that the number of colors used by the edges having one end in A and the other in B is minimized.

For ease, let $\texttt{col} : E \mapsto \mathcal{C}$ be a function returning the color of a given edge and $\texttt{mul}(c) = |\{e : e \in E \text{ and } \texttt{col}(e) = c\}|$ be a function returning the multiplicity of a given color. Moreover, for sake, given a graph $G = (V, E)$, let $\texttt{col}(G) = \bigcup_{e \in E} \texttt{col}(e)$ denote the set of colors used in G. Let us now describe an algorithm inspired by the folklore Contraction Algorithm [12] used for solving the classical Minimum Cut problem (*i.e.* minimizing the number of edges having one end in A and the other in B) on uncolored graph by randomized algorithm.

As in [12], our Colored Contraction Algorithm uses a connected multigraph $G = (V, E)$ – that is an undirected graph that is allowed to have more then one edge between the same pair of vertices – which is moreover colored. The algorithm starts by choosing, uniformly at random, a color $c \in \texttt{col}(G)$ and contracting any edge $e \in E$ such that $\texttt{col}(e) = c$ (and thus all such edges). Contracting an edge $(u, v) \in E$ will produce a new graph $G' = (V', E')$ in which u and v are identified as a single new vertex w whereas all other vertices are keeping their original identity (*i.e.* $V' = \{V \cup \{w\}\} \setminus \{u, v\}$). In G', $E' = \{E \cup \{(w, v'') : v' \in \{u, v\}, (v', v'') \in E\}\} \setminus \{(v', v'') : v' \in \{u, v\}, v'' \in V\}$. Roughly, E' is a copy of E where any edge (u, v) has been removed whereas any other edge has been preserved, but if one of its ends was equal to u or v, then this end is updated to be equal to the new node w. Note that the contraction operation may end up in a multigraph even when starting from a classical graph G. In this process, contracting all the edges that have the selected color c roughly corresponds to a sequence of $\texttt{mul}(c)$ contractions, each reducing the number of vertices by one. Colored Contraction Algorithm then continues recursively on G', by choosing, uniformly at random, a color $c \in \texttt{col}(G')$ and contracting any edge $e \in E$ such that $\texttt{col}(e) = c$. As these recursive calls proceed, the vertices of V' should be viewed as *supervertices*: each supervertex w corresponds to the subset $\mathcal{S}(w) \subseteq V$ that has been "swallowed up" in the contractions that produced w. The algorithm ends when it reaches a graph G'

with only two super-vertices v_A and v_B. We output $(A = \mathcal{S}(v_A), B = \mathcal{S}(v_B))$ as the colored-cut found by the algorithm.

Let us now analyze the performance of the COLORED CONTRACTION ALGORITHM – which cannot be derived directly from the one of the original CONTRACTION ALGORITHM. Since the algorithm is making random choices, there is some probability that it will succeed in finding a minimum colored-cut (and some probability that it would not). In order to prove that this algorithm is worthwhile, we will prove that the probability of success is only polynomially small; inducing that, by running the algorithm a polynomial number of times and returning the best colored-cut found in any run, one would be able to produce an optimal colored-cut with high probability.

Theorem 2. *The* COLORED CONTRACTION ALGORITHM *returns an optimal colored-cut G with probability at least $(|V|^{2k})^{-1}$ where $k = \text{MAX}_{c \in \mathcal{C}} \texttt{mul}(c)$*

Proof. Let us assume that the optimal minimum colored-cut (A, B) of G is of size OPT; that is the set of edges having one end in A and the other end in B (referred afterwards as the *cut-set*) is colored using OPT colors of $\mathcal{C}$. Note that unlike the classical MINIMUM CUT problem, the goal here is to minimize the number of colors in the cut-set itself. Moreover, let $G_{\text{OPT}} = G[A \cup B, \{(u, v) : (u, v) \in E \text{ and } u \in A, v \in B\}]$ corresponds to the bipartite graph representing the cut-set of (A, B). In order to compute a lower bound on the probability that the COLORED CONTRACTION ALGORITHM returns the minimum colored-cut (A, B), we first notice some important properties.

First, remark that any vertex $v \in V$ cannot have a degree less than OPT. Indeed, otherwise, $(\{v\}, V \setminus \{v\})$ would correspond to a colored-cut inducing at most $\text{OPT} - 1$ colors, contradicting our hypothesis that (A, B) is an optimal minimum colored-cut of G. Therefore, any vertex of G is of degree at least OPT; inducing the following lower bound on E: $|E| \geq \frac{\text{OPT}|V|}{2}$. We know moreover that, since each color of $\mathcal{C}$ can be used at most $k = \text{MAX}_{c \in \mathcal{C}} \texttt{mul}(c)$ times in E, we have that $|E| \leq k \cdot |\mathcal{C}|$. This leads to the following inequalities.

$$|V| \cdot \text{OPT} \leq 2 \cdot |E| \leq 2k \cdot |\mathcal{C}| \tag{1}$$

Let us now evaluate the probability $Pr[F_j]$ that the COLORED CONTRACTION ALGORITHM fails at the j^{th} step of the recursion (that is when already $j - 1$ contractions have been done). Considering what could go wrong in the j^{th} step of the COLORED CONTRACTION ALGORITHM, one can check that the unique issue would be that the uniformly at random choice of a color c unfortunately select one color of the set of OPT colors used by the cut-set – which will be then contracted inducing that the algorithm would not be able to find the optimal colored-cut (A, B) since at least a node of A and a node of B would be both contracted into the same supervertex. Hence the probability that an edge of the current graph G' is both in the optima cut-set and contracted is at most $\frac{\text{OPT}}{|\mathcal{C}'|}$, since there are at most OPT edges to be chosen among $|\mathcal{C}'|$ edges, where

$\mathcal{C}' = \texttt{col}(G')$. According to Inequality 1, considering that the graph at j^{th} step is G' and $\mathcal{C}' = \texttt{col}(G')$

$$Pr[F_j] \leq \frac{\text{OPT}}{|\mathcal{C}'|} \leq \frac{2k \cdot |\mathcal{C}'|}{|V'| \cdot |\mathcal{C}'|} = \frac{2k}{|V'|} \tag{2}$$

The colored-cut (A, B) will actually be returned by the algorithm if no edge of the cut-set is contracted in any of the at most $|V| - 2$ iterations. If we write S_j for the event that an edge of the cut-set has not been contracted until the j^{th} step, then, according to Inequality 2, $Pr[S_j] \geq 1 - Pr[F_j] = 1 - \frac{2k}{|V'|}$ where the graph at j^{th} step is $G' = (V', E')$. For ease, let us consider the sequence of color choices as being $\mathcal{S}_c = (c_1, c_2 \ldots)$ and $\lambda_j = \sum_{i<j \text{ and } c_i \in \mathcal{S}_c} \texttt{mul}(c_i)$. On the whole the probability that the Colored Contraction Algorithm returns the optimal colored-cut (A, B) is thus at least

$$Pr[Success] \geq \prod_{i=0}^{\lambda_1 - 1} (1 - \frac{2k}{|V| - i}) \cdot \prod_{i=\lambda_2}^{\lambda_3 - 1} (1 - \frac{2k}{|V| - i}) \ldots \prod_{i=\lambda_{|\mathcal{S}_c|-1}}^{\lambda_{|\mathcal{S}_c|} - 1} (1 - \frac{2k}{|V| - i}) \tag{3}$$

$$\geq \prod_{i=0}^{\lambda_1 - 1} (\frac{|V| - i - 2k}{|V| - i}) \cdot \prod_{i=\lambda_2}^{\lambda_3 - 1} (\frac{|V| - i - 2k}{|V| - i}) \ldots \prod_{i=\lambda_{|\mathcal{S}_c|-1}}^{\lambda_{|\mathcal{S}_c|} - 1} (\frac{|V| - i - 2k}{|V| - i}) \tag{4}$$

$$\geq \prod_{i=0}^{\lambda_{|\mathcal{S}_c|} - 1} (\frac{|V| - i - 2k}{|V| - i}) = \frac{\cancel{|V| - 2k}}{|V|} \ldots \frac{|V| - 2k - 2k}{\cancel{|V| - 2k}} \ldots \frac{|V| - (\lambda_{|\mathcal{S}_c|} - 1) - 2k}{\cancel{|V| - (\lambda_{|\mathcal{S}_c|} - 1)}} \tag{5}$$

$$\geq \frac{\prod_{i=2k}^{\lambda_{|\mathcal{S}_c|} - 1} |V| - i - 2k}{\prod_{i=0}^{2k-1} |V| - i} \geq \frac{1}{|V|^{2k}} = (|V|^{2k})^{-1} \tag{6}$$

□

Then according to Theorem 2, we know that a single run of the Colored Contraction Algorithm fails to find an optimal colored-cut with probability at most $(1 - (|V|^{2k})^{-1})$. One can then amplify the probability of success simply by repeatedly running the algorithm, with independent random choices, and taking the best colored-cut found. It is known that the function $(1 - n^{-1})^n$ converges monotonically from $\frac{1}{4}$ up to $\frac{1}{e}$ as n increases from 2 [12]. Thus, if we run the algorithm $|V|^{2k}$ times, then the probability that we fail to find an optimal colored-cut in any run is at most $(1 - (|V|^{2k})^{-1})^{|V|^{2k}} \leq \frac{1}{e}$. As usually done, it is easy to even reduce more the failure probability with further repetitions by running the algorithm $|V|^{2k} \ln |V|$ times which induces a probability of failure of at most $e^{-\ln |V|} = \frac{1}{|V|}$. On the overall, the running time required to get a high probability of success is polynomial in $|V|$, since each run of the Colored

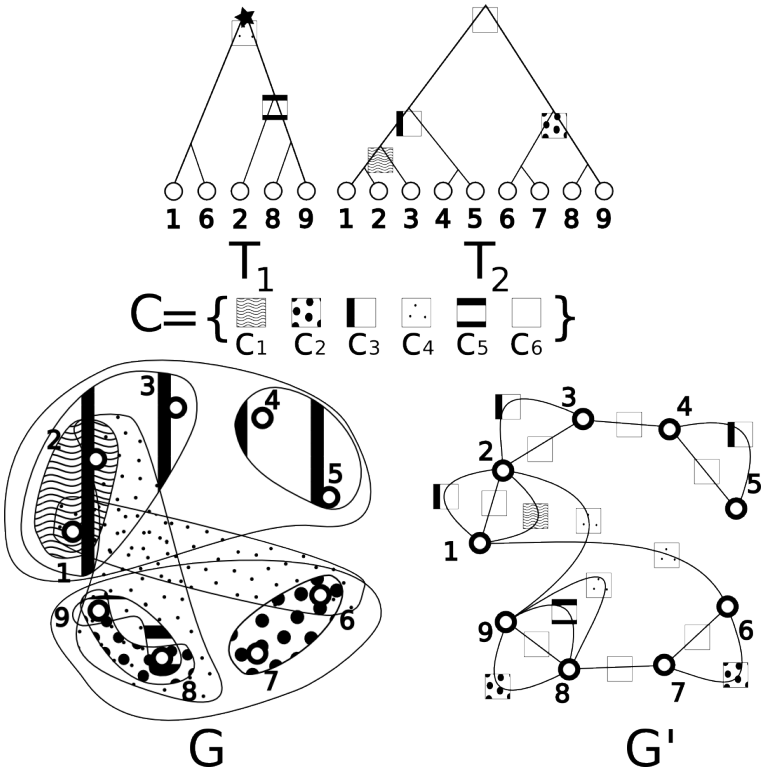

Fig. 4. Illustration of the construction of $G_{\mathcal{F}}$ and G' given $\mathcal{F} = (T_1, T_2)$. Considering the minimum colored-cut $\{1, 2, 3, 4, 5\}, \{6, 7, 8, 9\}$ of size 1, the only induced duplication is represented as a star on T_1.

Contraction Algorithm takes polynomial time, and we run it a polynomial number of times.

Let us now demonstrate how this result can be used in order to solve the Minimum Duplication Bipartite problem.

Theorem 3. *The* Minimum Duplication Bipartite *problem is randomized polynomial time solvable when the gene trees are of bounded depth.*

Proof. In the following, for ease, given a binary tree $T = (V, E)$ and a vertex $v \in V$, let us denote by v^L (resp. v^R) the left (resp. right) child of v and by ζ_v the cluster of v *i.e.* the set of all leaves belonging to the subtree rooted in v. Moreover, for ease, ϑ_T will denote the root of the tree T. Given a gene tree forest $\mathcal{F} = \{T_1 = (V_1, E_1), T_2 = (V_2, E_2), \ldots\}$ built on Λ, considering the definition of the Minimum Duplication Bipartite problem, one wants to define a bipartition (Λ_1, Λ_2) of $\Lambda = \bigcup_{T_i \in \mathcal{F}} V_i$ inducing the minimum number of

pre-duplications. In T_i, a node v of V_i is a duplication with respect to (Λ_1, Λ_2), if $\exists v' \in \{v^L, v^R\}$, such that $(\Lambda_1 \bigcap \zeta_{v'} \neq \emptyset) \wedge (\Lambda_2 \bigcap \zeta_{v'} \neq \emptyset)$ is true. In other words, v is a duplication if for one of its children – say v' – $\zeta_{v'}$ contains two leaves not belonging to the same part of the bipartition (Λ_1, Λ_2). Given $\mathcal{F}$ and a set of colors $\mathcal{C}$, we define the following colored hypergraph $G_{\mathcal{F}} = (V, E)$ associated to $\mathcal{F}$. Let $V = \Lambda = \bigcup_{T \in \mathcal{F}} \zeta_{\vartheta_T}$ and there are two hyperedges, for any node v_k of the tree T_i, $\alpha_k^i = \{\zeta_{v_k^L} : |\zeta_{v_k^L}| \geq 2\}$ and $\beta_k^i = \{\zeta_{v_k^R} : |\zeta_{v_k^R}| \geq 2\}$ colored with color $\texttt{col}(\alpha_k^i) = \texttt{col}(\beta_k^i) = c_k^i \in \mathcal{C}$ in E. An illustration of such construction is provided in Fig. 4. Then in $G_{\mathcal{F}}$, a colored-cut of size k' corresponds to a bipartition of the set Λ inducing k' duplications. Indeed, if the hyperedge α_k^i (resp. β_k^i) belongs to the cut-set, then it induces a duplication for the corresponding vertex v_k in T_i since there exist at least two leaves in $\zeta_{v_k^L}$ (resp. $\zeta_{v_k^R}$) belonging to different parts of the bipartition (Λ_1, Λ_2).

Thus, if one can find a minimum colored-cut in such hypergraphs, then one would be able to solve in polynomial time the Minimum Duplication Bipartite problem. Just consider the Colored Contraction Algorithm presented previously in this section. From any colored hypergraph $G_{\mathcal{F}} = (V, E)$, one may build a colored graph $G' = (V, E')$ where any hyperedge $e = \{v_{i1}, v_{i2} \dots v_{ik}\}$ colored with color $c = \texttt{col}(e)$ has been replaced by a path $v_{i1}, v_{i2} \dots v_{ik}$ colored with c in E' (*i.e.* $E' = \{(v_{ik}, v_{ik+1}) : v_{ik} \in e, e \in E\}$). Notice that an edge $e \in E'$ colored with c is cut if and only if an hyperedge colored c of $G_{\mathcal{F}}$ is cut. Once this colored graph has been obtained, one may apply the Colored Contraction Algorithm which will produce a minimum colored-cut of G' which also induces a minimum colored cut in $G_{\mathcal{F}}$. Since this algorithm has a complexity exponential in the maximum multiplicity of any color of the considered graph, when the size of each hyperedge is bounded, so does the multiplicity of any color since the maximal size of an hyperedge corresponds to the maximal depth of the input gene trees: leading to a randomized polynomial solution for the Minimum Duplication Bipartite problem. □

4 Conclusion

In this paper we have investigated the complexity of two variants of the Minimum Duplication problem. We have proved that the Minimum Duplication problem is APX-hard, even when the input consists of five uniquely leaf-labelled gene trees. Then, we have shown that the Minimum Duplication Bipartite problem can be solved efficiently by a randomized algorithm when the input gene trees have bounded depth.

A natural open problem is the complexity of the Minimum Duplication Bipartite problem when the gene trees have unbounded depth. Furthermore, it would be interesting to deepen the analysis on the complexity of the Minimum Duplication problem, when the input consists of less than five uniquely leaf-labelled gene trees.

References

1. P. Alimonti and V. Kann. Some APX-completeness results for cubic graphs. *Theoretical Comput. Sci.*, 237(1–2):123–134, 2000.
2. M. S. Bansal, J. G. Burleigh, O. Eulenstein, and A. Wehe. Heuristics for the Gene-Duplication Problem: A $\Theta(n)$ -Speed-Up for the Local Search. In T. P. Speed and H. Huang, editors, *RECOMB*, volume 4453 of *LNCS*, pages 238–252. Springer, 2007.
3. M. S. Bansal, O. Eulenstein, and A. Wehe. The Gene-Duplication Problem: Near-Linear Time Algorithms for NNI-Based Local Searches. *IEEE/ACM Trans. Comput. Biology Bioinform.*, 6(2):221–231, 2009.
4. M. S. Bansal and R. Shamir. A Note on the Fixed Parameter Tractability of the Gene-Duplication Problem. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 8(3):848–850, 2011.
5. J. Byrka, S. Guillemot, and J. Jansson. New results on optimizing rooted triplets consistency. *Discrete Appl Math*, 158(11):1136–1147, 2010.
6. W.-C. Chang, J. G. Burleigh, D. F. Fernández-Baca, and O. Eulenstein. An ILP solution for the gene duplication problem. *BMC Bioinformatics*, (Suppl 1):S14(12), 2011.
7. C. Chauve and N. El-Mabrouk. New Perspectives on Gene Family Evolution: Losses in Reconciliation and a Link with Supertrees. In S. Batzoglou, editor, *RECOMB*, volume 5541 of *LNCS*, pages 46–58. Springer, 2009.
8. E. E. Eichler and D. Sankoff. Structural dynamics of eukaryotic chromosome evolution. *Science*, 301(5634):521–565, 2003.
9. J. Felsenstein. Phylogenies from molecular sequences: Inference and reliability. *Ann. Review Genet.*, 22:521–565, 1988.
10. W. M. Fitch. Homology—a personal view on some of the problems. *Trends Genet.*, 16:227–231, 2000.
11. M. T. Hallett and J. Lagergren. New algorithms for the duplication-loss model. In *RECOMB*, pages 138–146, 2000.
12. J. Kleinberg and E. Tardos. *Algorithm Design.* Pearson Education, 2006.
13. B. Ma, M. Li, and L. Zhang. From Gene Trees to Species Trees. *SIAM J. Comput.*, 30(3):729–752, 2000.
14. A. Ouangraoua, K. M. Swenson, and C. Chauve. An Approximation Algorithm for Computing a Parsimonious First Speciation in the Gene Duplication Model. In E. Tannier, editor, *RECOMB-CG*, volume 6398 of *LNCS*, pages 290–301, Ottawa, Canada, Oct. 2010. Springer.
15. U. Stege. Gene Trees and Species Trees: The Gene-Duplication Problem in Fixed-Parameter Tractable. In F. K. H. A. Dehne, A. Gupta, J.-R. Sack, and R. Tamassia, editors, *6th International Workshop on Algorithms and Data Structures (WADS'99)*, volume 1663 of *LNCS*, pages 288–293. Springer, 1999.
16. D. J. A. Welsh and M. B. Powell. An upper bound for the chromatic number of a graph and its application to timetabling problems. *The Computer Journal*, 10(1):85–86, 1967.

5 Appendix - Detailed proof of APX-hardness

5.1 Extra notations

Given a binary tree $T = (V, E)$, with leaves labelled by Λ and $\Lambda' \subseteq \Lambda$, we define the *restriction of* T *to* Λ', denoted $T|\Lambda'$, as the subtree obtained from T by retaining only leaves with a label belonging to Λ' and by contracting all the internal vertices of degree 2. For ease, we will note $\texttt{lca}_T(u, v)$ the lowest common ancestor of two nodes u and v in a tree T.

For ease, in the following, we will consider that $n = |V_G|$ and $m = |E_G|$. Let us define the following ordered sequences of labels:
- $L_f = \langle f_{n+1}, f_n^1, \ldots, f_n^q, f_n, \ldots f_1^1, \ldots, f_1^q, f_1 \rangle$;
- $L_f^* = \langle f_n^1, \ldots, f_n^q, \ldots, f_1^1, \ldots, f_1^q \rangle$;
- $L_f' = \langle f_{n+1}, f_n, \ldots, f_1 \rangle$.
- $L_f^i = \langle f_i^1, \ldots, f_i^q \rangle$, with $1 \leq i \leq n$.

Moreover, let P_x be the set of internal vertices in T_x, $1 \leq x \leq 4$, belonging to the path from the root of T_x to the parent p_{n+1}^x of f_{n+1}. We define the *spine of any gene tree* T_x, $1 \leq x \leq 4$, as $P_x \setminus \{p_{n+1}^x\}$.

Recall that given a gene tree T and a species tree S, a vertex v of T is duplicated with respect to S if $\mathcal{M}(v) = \mathcal{M}(v')$ where v' is a descendant of v in T (*i.e.* v' belongs to T_v).

5.2 Preliminary properties

Let us introduce some fundamental properties that will be used in the rest of this appendix.

Property 1. Let T, T' be two gene trees labelled by the same sets of leaves Λ. Consider the bipartitions $b_1 = (\zeta_{\vartheta_T^L}, \zeta_{\vartheta_T^R})$, $b_2 = (\zeta_{\vartheta_{T'}^L}, \zeta_{\vartheta_{T'}^R})$ of Λ. Then either b_1 and b_2 are identical or any species tree S induces at least one duplication in the root of one of $\{T, T'\}$.

Property 2. Let T be a gene tree and S be a species tree labelled by the same sets of leaves Λ. Then either T and S are isomorphic or $d(T, S) \geq 1$.

Proof. Let us prove this lemma by induction. By hypothesis, let T and S be two trees not isomorphic. If both T and S have depth 2, it is easy to check that $d(T, S) \geq 1$. Assume now that T and S have depth larger than 2. Then at least one of the following statements holds: (a) $T|\zeta_{\vartheta_T^L}$ and $S|\zeta_{\vartheta_S^L}$ are not isomorphic or (b) $T|\zeta_{\vartheta_T^R}$ and $S|\zeta_{\vartheta_S^L}$ are not isomorphic or (c) $T|\zeta_{\vartheta_T^R}$ and $S|\zeta_{\vartheta_S^L}$ are not isomorphic or (d) $T|\zeta_{\vartheta_T^R}$ and $S|\zeta_{\vartheta_S^R}$ are not isomorphic. Then, by induction, $d(T, S) \geq 1$. □

Property 3. Let $T = (V_T, E_T)$ be a gene tree and $S = (V_S, E_S)$ be a species tree. Let v be a vertex of V_T such that v has at least one child v', which is not a leaf. If there exists a vertex w of V_S such that (a) $\zeta_v \setminus \zeta_{v'} \subseteq \zeta_w$, (b) $\zeta_w \not\supseteq \zeta_v$ and (c) $\zeta_w \cap \zeta_{v'} \neq \emptyset$, then v is duplicated.

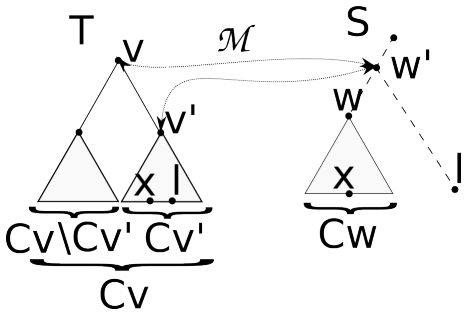

Fig. 5. Illustration of Property 3 where (a) $\zeta_v \setminus \zeta_{v'} \subseteq \zeta_w$, (b) $\zeta_w \not\supseteq \zeta_v$ and (c) $\zeta_w \cap \zeta_{v'} \neq \emptyset$.

Proof. Let us consider the vertex w in S. Notice that, since (a) $\zeta_v \setminus \zeta_{v'} \subseteq \zeta_w$, (b) $\zeta_w \not\supseteq \zeta_v$ and (c) $\zeta_w \cap \zeta_{v'} \neq \emptyset$ then there exist at least a label l, such that $l \in \zeta_{v'} \setminus \zeta_w$ (otherwise ζ_w would contain ζ_v). Furthermore, as $\zeta_w \cap \zeta_{v'} \neq \emptyset$, it follows that v' and v are mapped with vertices of S that are on the path from w to ϑ_S. Let w' be the vertex of S to which v' is mapped (*i.e.* $\mathcal{M}(v') = w'$). Note that w' is defined such that $\zeta_{v'} \subseteq \zeta_{w'}$ and $\nexists z$ such that $\zeta_{v'} \subseteq \zeta_z$ and $|\zeta_z| < |\zeta_{w'}|$. Since w' is an ancestor of w, it follows that $\zeta_v \subseteq \zeta_{w'}$. Hence, v is mapped with w' (*i.e.* $\mathcal{M}(v) = w'$). As a consequence v is duplicated. □

5.3 Missing proofs

First of all, let us prove that, by construction, all the gene trees $T_1, \ldots, T_5$ are uniquely leaf-labelled.

Lemma 2. *The trees $T_1, \ldots, T_5$ are uniquely leaf-labelled trees.*

Proof. It is easy to see that T_5 is uniquely leaf-labelled by construction. Indeed, each of the three comb graphs $C(L_f)$, $C(M_1)$, $C(M_2)$ used to build T_5 is uniquely leaf-labelled. Moreover, the trees $C(L_f)$, $C(M_1)$, $C(M_2)$ have pairwise disjoint sets of leaves.

Now consider the gene trees T_1, T_2, T_3 and T_4. First, remark that any tree T_v with $v \in V_G$ is uniquely leaf-labelled and so do the trees $T^1_{e_x}$ and $T^2_{e_x}$. Then, one has only to pay attention to their relative placement in the gene trees. More precisely, by construction, one has to be sure that a tree T_v where $v \in V_G$ is incident to edges $e_x = (v, v')$, $e_y = (v, v'')$ and $e_z = (v, v''')$ does not belong to the same gene tree than none of $\{T_{v'}, T_{v''}, T_{v'''}\}$ and none of $\{T^1_{e_x}, T^2_{e_x}, T^1_{e_y}, T^2_{e_y}, T^1_{e_z}, T^2_{e_z}\}$. This is indeed true since all those trees are associated to the gene trees considering their corresponding color in the 4-coloring of G. One has also to be sure that the trees $T^1_{e_x}$ and $T^2_{e_x}$ do not belong to the same gene tree (which is the case by construction). □

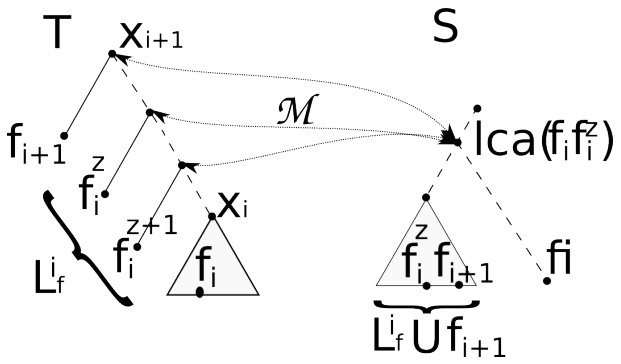

Fig. 6. Illustration of Lemma 3 when for each $f_i^z \in L_f^i$, $\texttt{lca}_S(f_i, f_i^z) > \texttt{lca}_S(f_{i+1}, f_i^z)$.

Let us now prove that only *canonical* solutions are of interest. Remind that a canonical solution (*i.e.* a species tree S) is a copy of T_5 where extra leaves of $L = \{l_x^1, l_x^2 : \forall e_x \in E_G\}$ are each inserted either in $C(M_1)$ or $C(M_2)$. More formally, a species tree S is *canonical* if and only if $S|(M_1 \cup M_2 \cup L_f)$ is isomorphic to T_5 and any label in $\zeta_S \setminus \zeta_{T_5}$ belongs to ζ_v where v is the other child of the parent of f_1 in S.

First, let us consider the following order induced by the `lca` mapping $\mathcal{M}$. Consider three vertices v, v', v'' of a tree T and the following ordering of their lowest common ancestors: we write $\texttt{lca}_T(v, v') > \texttt{lca}_T(v', v'')$ when the depth of the `lca` of v', v'' is greater than the one of v, v'. We will moreover note $\texttt{lca}_T(v, v') \geq \texttt{lca}_T(v', v'')$ or $\texttt{lca}_T(v', v'') \geq \texttt{lca}_T(v, v')$ when the depth of the `lca` of v, v' and the `lca` of v', v'' is equal.

Lemma 3. *Let S be a solution of* Min-5-Dup *for the instance $\mathcal{F} = \{T_1, \ldots, T_5\}$. Then, either $d(\mathcal{F}, S) \geq 6n + 3m + 1$ or all the vertices on the spines of the gene trees T_1, T_2, T_3, T_4 are duplicated.*

Proof. Consider any species tree S and two leaves f_i, f_{i+1} of T_5, for a given $1 \leq i < n + 1$. Let w_i^j (resp. w_{i+1}^j) be the parent of f_i (resp. f_{i+1}) in the gene tree T_j, with $j \in \{1, 2, 3, 4\}$. Let x_i (resp. x_{i+1}) denote the parent in T_5 of f_i (resp. f_{i+1}). Moreover, let y_1 (resp. y_2) be the vertices in S mapped, according to the `lca` mapping $\mathcal{M}$, with x_i (resp. x_{i+1}) – *i.e.* $\mathcal{M}(x_i) = y_1$ (resp. $\mathcal{M}(x_{i+1}) = y_2$).

In what follows, we consider a label f_i^z, with $1 \leq z \leq q$, and prove that, considering the previously mentioned mapping, either all the internal vertices on the path from x_i to x_{i+1} in T_5 are duplicated (hence $d(\mathcal{F}, S) \geq q = 6n + 3m + 1$) or all the internal vertices on the path from w_i^j (included) to w_{i+1}^j (not included) are duplicated in T_j, $1 \leq j \leq 4$. To do so, we will consider a case by case analysis based on the possible mappings of f_i, f_{i+1} and f_i^z in S.

Assume that for each $f_i^z \in L_f^i$, $\texttt{lca}_S(f_i, f_i^z) \geq \texttt{lca}_S(f_{i+1}, f_i^z)$. Notice that there exists two possible cases: $\texttt{lca}_S(f_i, f_i^z) > \texttt{lca}_S(f_{i+1}, f_i^z)$, for each f_i^z, or

there exists at least one f_i^z such that $\texttt{lca}_S(f_i, f_i^z) > \texttt{lca}_S(f_{i+1}, f_i)$. If $\texttt{lca}_S(f_i, f_i^z) > \texttt{lca}_S(f_{i+1}, f_i^z)$, for each f_i^z, then Property 3 applies to each internal vertex between x_i, x_{i+1}. Indeed for each internal vertex between x_i, x_{i+1}, there exists a vertex between $\texttt{lca}_S(f_i, f_i^z)$ and $\texttt{lca}_S(f_{i+1}, f_i^z)$ such that Property 3 applies, as the cluster of each of the latter vertices does not contain f_i (see Figure 6). Hence $d(\mathcal{F}, S) \geq q = 6n + 3m + 1$. Now assume that for some f_z^i, $\texttt{lca}_S(f_i, f_i^z) > \texttt{lca}_S(f_{i+1}, f_i)$. Let us consider the leaves in $\zeta_{w_i^x} \setminus \zeta_{w_{i+1}^x}$ and let S_i^x be the set of internal vertices of T_x between w_{i+1}^x and w_i^x. Property 3 applies to the vertex w_i^x, as $\texttt{lca}_S(f_{i+1}, f_i)$ contains f_i, f_{i+1} but not f_i^z, which is contained in both $\zeta_{w_{i+1}^x}$ and $\zeta_{w_i^x}$. Hence w_i^x is duplicated. Now, consider the lowest vertex $s_i^x \in S_i^x$ not duplicated and denote by t_i^x its child which is not on the spine of T_x, $1 \leq x \leq 4$. Let z_1 be the vertex of S where s_i^x is mapped (*i.e.* $\mathcal{M}(s_i^x) = z_1$), and notice that $z_1 \geq \texttt{lca}_S(f_i^z, f_i)$. Since s_i^x is not duplicated, then the cluster of one of the children of z_1 contains $\zeta_{t_i^x}$, while the other contains $\{f_i^z, f_i, f_{i+1}\}$, for each $1 \leq z \leq q$. But then, since $m_x^1, m_x^2 \in \zeta_{s_i^x}$, it follows that for each internal node between x_i, x_{i+1}, there exists a vertex between $\texttt{lca}_T(f_i, f_i^z)$ and z_1 such that Property 3 applies, as the cluster of each of the latter vertices does not contain m_x^2, while the cluster of each vertex between x_i and x_{i+1} contains a label m_x^2. Hence $d(\mathcal{F}, S) \geq 6n + 3m + 1$.

Now, let us consider the case $\texttt{lca}_S(f_i, f_i^z) < \texttt{lca}_S(f_{i+1}, f_i^z)$. It follows by Property 3 that the internal vertices of S_i^x and w_i^x are all mapped in a vertex z_1, hence inducing a duplication in w_i^x. Indeed, $\zeta_{w_i^x} = \zeta_{s_i^x} \setminus \{f_i\}$, while $\texttt{lca}_T(f_i, f_i^z)$ contains f_i, but not f_{i+1}.

Now consider the lowest vertex $s_i^x \in S_i^x$ that is not duplicated. Then the cluster of one of the children of z_1 (the vertex where s_i^x and w_i^x are mapped) contains $\zeta_{s_i^x}$, while the other contains $\{f_i^z, f_i, f_{i+1}\}$, for each $1 \leq z \leq q$. Since $m_x^2 \in \zeta_{s_i^x}$, it follows that for each internal node between x_i, x_{i+1}, there exists a vertex between $\texttt{lca}_S(f_i, f_i^z)$ and z_1 such that Property 3 applies, as the cluster of each of the latter vertices does not contain m_x^2, while the cluster of each vertex between x_i and x_{i+1} contains m_x^2. Hence $d(\mathcal{F}, S) \geq 6n + 3m + 1$.

Since we have shown that for each pair of leaves f_i, f_{i+1}, either $d(\mathcal{F}, S) \geq 6n + 3m + 1$, or all the internal nodes between w_i^x (included) and w_{i+1}^x (not included) are duplicated, it follows that we have proved the lemma. □

While in the previous lemma we have focused on the duplications induced on vertices of the spine of the gene trees $T_1, \ldots, T_4$, in what follows, we will focus on the duplications induced in the subtrees representing the vertices and edges (*i.e.* T_v, T_e^x).

Lemma 4. *Let S be a solution of* Min-5-Dup *over instance $\mathcal{F} = \{T_1, \ldots, T_5\}$ and let $T_{e_x}^1$, $T_{e_x}^2$, T_{v_i}, T_{v_j} be four subtrees of $T_1, \ldots, T_4$, s.t. $e_x = (v_i, v_j) \in E_G$. Then (1) the root of at least one of $T_{e_x}^1$, $T_{e_x}^2$ is duplicated with respect to S; (2) the roots of at least two of $T_{e_x}^1$, $T_{e_x}^2$, T_{v_i}, T_{v_j} are duplicated with respect to S.*

Proof. It follows from Property 1 that the root of at least one of $T_{e_x}^1$, $T_{e_x}^2$ is duplicated with respect to S since both $T_{e_x}^1$, $T_{e_x}^2$ are labelled by the same set of leaves and they are not isomorphic.

Now, let us prove the second part of the lemma. We have shown that any species tree induces a duplication in the root of at least one of $T^1_{e_x}$, $T^2_{e_x}$. Let us consider a species tree S. If S induces a duplication in the roots of both $T^1_{e_x}$ and $T^2_{e_x}$, then the lemma holds. Hence assume that S induces a duplication in exactly one of $T^1_{e_x}$, $T^2_{e_x}$, w.l.o.g. $T^1_{e_x}$. Thus, assume that S does not induce a duplication in the root of $T^2_{e_x}$.

Let us define $L_x = \{m^1_x, m^2_x, l^1_x, l^2_x\}$ and consider the following restrictions of T_{v_i} and T_{v_j}: $T_{v_i}|L_x$, $T_{v_j}|L_x$. The roots of both $T_{v_i}|L_x$ and $T_{v_j}|L_x$ induce the following bipartitions $B(v_i) = (\{m^1_x, l^1_x\}; \{m^2_x\})$ and $B(v_j) = (\{m^1_x, l^2_x\}; \{m^2_x\})$.

Let v be the vertex of S, which is the lowest common ancestor of $\{m^2_x, l^1_x, l^2_x\}$. Since we have assumed that the root of $T^2_{e_x}$ is not duplicated, it follows that the subtree rooted at v restricted to $\{m^2_x, l^1_x, l^2_x\}$ must induce the bipartition $(\{m^2_x, l^1_x\}; \{l^2_x\})$. Now assume that both the root of T_{v_i} and the root of T_{v_j} are mapped to v and consider where the leaf m^1_x is possibly placed in a the subtree rooted in v. If m^1_x is clustered with l^2_x, then the root of T_{v_j} is duplicated, as bipartition $B(v_j) = (\{m^1_x, l^2_x\}; \{m^2_x\})$. If m^1_x is clustered with l^1_x and m^2_x, then the root of T_{v_i} is duplicated, as bipartition $B(v_i) = (\{m^1_x, l^1_x\}; \{m^2_x\})$.

Assume now that the root of T_{v_i} or the root of T_{v_j}, w.l.o.g. $\vartheta_{T_{v_i}}$, is not mapped to v. Then, both the root of T_{v_i} and at least one of its children are mapped to an ancestor of v. Let y be the ancestor of v to which $\vartheta_{T_{v_i}}$ is mapped. The clusters of both children of y contain a leaf of $\zeta_{\vartheta_{T_{v_i}}}$. More precisely, the cluster of the child z of y that is on the path from v to y will contain the leaf set $\{m^1_x, l^1_x, m^2_x\}$, as so does the ζ_v. Hence, one of the children of $\vartheta_{T_{v_i}}$, w.l.o.g. $\vartheta^L_{T(V_i)}$, has a cluster not included in ζ_z but sharing some elements with ζ_z. This implies that $\vartheta^L_{T(V_i)}$ is mapped in y, thus a duplication occurs in the root of T_{v_i}. □

Let us now consider canonical solution and prove the following lemma.

Lemma 5. *Let S be a solution of* Min-5-Dup *over instance $\mathcal{F} = \{T_1, \ldots, T_5\}$. Then, we can compute in polynomial time a canonical solution S^* of* Min-5-Dup *over instance $\mathcal{F}$, such that $d(\mathcal{F}, S^*) \leq d(\mathcal{F}, S)$, and such that for each $e_x = (v_i, v_j) \in E_G$ and the corresponding subtrees $T^1_{e_x}$, $T^2_{e_x}$, T_{v_i}, T_{v_j} of T_1, T_2, T_3, T_4: (1) the root of exactly one of $T^1_{e_x}$, $T^2_{e_x}$ is duplicated with respect to S^*; (2) the root of at least one of T_{v_i}, T_{v_j} is duplicated with respect to S^*.*

Proof. Let S be a solution of Min-5-Dup over instance $T_1, \ldots, T_5$. If S induces at least $6n + 3m + 1$ duplications, since a canonical solution induces at most $5n + 3m + p$ duplications, where $p \leq n$, it follows that for any canonical solution S^* that satisfies properties (1) and (2) of the lemma, $d(\mathcal{F}, S^*) \leq d(\mathcal{F}, S)$. Indeed in a canonical solution S^*, there are $5n + 2m$ duplications on the spines of the gene trees T_1, T_2, T_3, T_4, plus m duplications in the root of either $T^1_{e_x}$ or $T^2_{e_x}$, for any $e_x \in E_G$ and p duplications in the roots of any subtree T_{v_i}, for any $v_i \in V_G$.

Hence, assume that S induces less than $6n+3m+1$ duplications. By Lemma 3, it follows that S induces a duplication in all the the vertices on the spine of each gene tree T_i, $1 \leq i \leq 4$. Now, consider the set R of clusters associated with the roots of subtrees $T^1_{e_x}$, $T^2_{e_x}$, T_{v_i}, T_{v_j} and define two subsets of R as follows: $Safe(S) = \{x \in R : x \text{ is not duplicated in } S\}$, $Dup(S) = R \setminus Safe(S)$.

Now let us compute a canonical solution S^*. First, starting from $Safe(S)$ and $Dup(S)$, let us compute two sets $Safe(S^*)$ and $Dup(S^*)$ as follows. Notice that by Lemma 4, at least one of the roots of $T^1_{e_x}$, $T^2_{e_x}$ is duplicated. We assume w.l.o.g. that if the cluster of $\vartheta_{T_{v_i}}$ (resp. of $\vartheta_{T_{v_j}}$) is in $Dup(S)$, then the cluster of $\vartheta_{T^1_{e_x}}$ (resp. of $\vartheta_{T^2_{e_x}}$) is in $Dup(S)$. Notice that, if the clusters of both $\vartheta_{T^1_{e_x}}$, $\vartheta_{T^2_{e_x}}$ are in $Dup(S)$, then we can assign one of these two clusters to $Safe(S)$ and eventually one of $\vartheta_{T_{v_i}}$, $\vartheta_{T_{v_j}}$ to $Dup(S)$, without improving the size of $Dup(S)$. Now, define $Safe(S^*) = Safe(S)$ and $Dup(S^*) = Dup(S)$.

Let S' be a tree isomorphic to T_5. Then starting from S' we construct a solution S^* of Min-5-Dup as follows. Let us consider the subtree of S' having as leaf set $M_1 \cup M_2$. Let r' be the root of this subtree, and let r_i, $1 \leq i \leq 2$, be the root of $C(M_i)$ in S'.

First, let us assign the elements of $L = \langle l^1_1, l^2_1, l^1_2, l^2_2, \ldots l^1_{|E_G|}, l^2_{|E_G|} \rangle$ to two sets L_1, L_2 as follows. If the cluster of $\vartheta_{T^1_{e_x}}$ (resp. $\vartheta_{T^2_{e_x}}$) belongs to $Safe(S^*)$, then assign the leaves l^1_x (resp. l^2_x) to L_1. If the cluster of $\vartheta_{T_{v_i}}$ belongs to $Safe(S^*)$, then assign l^1_x, l^1_y, l^1_z to L_1 where v_i is incident to the edges e_x, e_y and e_z. All the other leaves in L are assigned to L_2. Then a subtree $C(L_1)$ is inserted in the edge $\{r', r_1\}$. The subtree $C(M_2)$ is substituted with a subtree $C(M_2 \cup L_2)$, where the order on $M_2 \cup L_2$ is induced by the order on the corresponding edges and that $l^1_x < m^2_x$.

Now, we claim that the tree S^* induces at most $Dup(S^*)$ duplications, for the trees $T^1_{e_x}$, $T^2_{e_x}$, T_{v_i}, T_{v_j}. Indeed, assume that $T^1_{e_x}$ is in $Safe(S^*)$. Then, since at most one of l^1_x, l^2_x is assigned to L_1, at least one of l^1_x and l^2_x is assigned to L_2. It follows that the root of $T^1_{e_x}$ is not duplicated (a similar proof holds for $T^2_{e_x}$).

Now consider a subtree T_{v_i} in $Safe(S^*)$. As l^1_x, m^1_x are assigned to L_1, and as m^2_x is assigned to L_2, it follows that the root of T_{v_i} is not duplicated. Since the other vertices are arranged following the order on L, it follows that no duplication is induced in any internal vertex of T_{v_i}.

Now consider a subtree T_{v_i} in $Dup(S^*)$. By construction, the subtrees of $T_{v_i}|M_1$ and $T_{v_i}|M_2$ do not have duplications. Furthermore, as all the leaves l^1_x, l^1_y, l^1_z (related to v_i) are assigned to L_2, it follows that also the vertices in the restriction $T_{v_i}|\{l^1_x, l^1_y, l^1_z\}$ are not duplicated. Finally, the vertex of T_{v_i} associated with clusters $\{m^1_x, m^1_y, m^1_z, l^1_x, l^1_y, l^1_z\}$ is mapped to r', hence it is not duplicated as its children are not mapped to r'.

Now, consider the other vertices of T_1, T_2, T_3, T_4 not yet considered. Since by construction, the only other vertices of T_1, T_2, T_3, T_4 that are duplicated with respect to S^* are those vertices in the spines of the gene trees T_1, T_2, T_3, T_4, the correctness of the lemma follows. □

We are now ready to prove the main lemma; that is that our reduction is an L-reduction.

Lemma 6. *Let $G = (V_G, E_G)$ be an instance of* MVCC *and let $\mathcal{F} = \{T_1, \ldots, T_5\}$ be the corresponding instance of* Min-5-Dup. *Then, starting from a cover V'_G of G, we can compute in polynomial time a solution of* Min-5-Dup *over instance $\mathcal{F} = \{T_1, \ldots, T_5\}$ such that $d(\mathcal{F}, S) \leq 5n + 3m + |V'_G|$. Moreover, starting*

from a solution S to Min-5-Dup *over instance $\mathcal{F} = \{T_1, \ldots, T_5\}$ such that $d(\mathcal{F}, S) \leq 5n + 3m + p$, we can compute in polynomial time a cover of G of size at most p.*

Proof. Given a cover V'_G of $G = (V_G, E_G)$, define a solution S to Min-5-Dup as follows. The construction follows that of Lemma 5, by assigning T_{v_i} to $Safe(S)$ if $v_i \notin V'$.

Let us consider a species tree S inducing at most $5n + 3m + p$ duplications. By Lemma 5, it follows that starting from S, we can compute in polynomial time a canonical solution S^* of Min-5-Dup over instance $\mathcal{F} = \{T_1, \ldots, T_5\}$, inducing $5n + 3m + p'$ duplications, with $p' \leq p$. Since S^* is a canonical solution, we can assume that there exists exactly p' duplications induced by S^* in the subtrees $\{T_{v_i} : v_i \in V_G\}$. By Lemma 5, for each edge $\{v_i, v_j\} \in E_G$, at least one of the subtrees from $\{T_{v_i}, T_{v_j}\}$ has a duplication in its root. Hence, the set of vertices $V' \subseteq V$ corresponding to those trees in $\{T_{v_i} : v_i \in V_G\}$ having a duplication in their root forms a cover of G. Since $|V'| = p' \leq p$, it follows that the lemma holds. □