Detecting Locus Acquisition Events in Gene Trees*

Michał Aleksander Ciach[1], Anna Muszewska[2], and Paweł Górecki[3]

1 **Faculty of Mathematics, Informatics, and Mechanics, University of Warsaw, Warsaw, Poland; and**
Institute of Biochemistry and Biophysics, Polish Academy of Sciences, Warsaw, Poland
m_ciach@student.uw.edu.pl
2 **Institute of Biochemistry and Biophysics, Polish Academy of Sciences, Warsaw, Poland**
3 **Faculty of Mathematics, Informatics, and Mechanics, University of Warsaw, Warsaw, Poland**
gorecki@mimuw.edu.pl

Abstract

Horizontal Gene Transfer (HGT), a process of acquisition and fixation of foreign genetic material, is an important biological phenomenon. Several approaches to HGT inference have been proposed. However, most of them either rely on approximate, non-phylogenetic methods or on the tree reconciliation, which is computationally intensive and sensitive to parameter values. In this work, we investigate the Locus Tree Inference problem as a possible alternative that combines the advantages of both approaches. We show several algorithms to solve the problem in the parsimony framework. We introduce a novel tree mapping, which allows us to obtain a heuristic solution to the problems of locus tree inference and duplication classification. Our approach allows not only for faster comparisons of gene and species trees but also to improve known algorithms for duplication inference in the presence of polytomies in the species trees.

1998 ACM Subject Classification J.3 [Life and Medical Sciences] Genomics

Keywords and phrases rank, taxon, ranked species tree, speciation, gene duplication, gene loss, horizontal gene transfer

Digital Object Identifier 10.4230/LIPIcs.WABI.2017.5

1 Introduction

Horizontal Gene Transfer (HGT) is the process of acquisition and fixation of foreign genetic material. It can lead to substantial changes in the ecology and evolution of recipient organism, sometimes leading to the emergence of new pathogens [8]. HGT is interesting both from biological and computational perspective. Several methods of detecting horizontally transferred genes have been proposed, which can be roughly divided into two categories [14]. So-called *surrogate methods* are computationally efficient, yet often imprecise. The other group are the the *phylogenetic methods*, most notably the tree reconciliation [5].

HGT and gene duplication are examples of evolutionary events in which an organism gains a new locus, i.e. a fragment of a chromosome with a specific gene. The new locus evolves more or less independently of other loci. This observation leads to the concept of a *locus tree* [13],

* The support was provided by NCN grants #2015/19/B/ST6/00726 and 2012/07/D/NZ2/04286.

which represents the evolutionary history of the loci. A corresponding gene tree represents evolutionary histories of alleles found in those loci, including populational effects like the incomplete lineage sorting. Therefore, a locus tree is an "intermediate" concept between the gene and the species tree. When population effects are negligible, a locus tree is equivalent to a gene tree with some branches labelled as locus gain events. Such labelling allows to "decompose" the gene tree into a set of independent evolutionary histories of different loci. The concept of gene tree decomposition has been investigated earlier in the context of tree comparison [10]. Distinguishing between different locus gain events is challenging, as their effects on gene trees are topologically similar. In reconciliation, weights of events have to be specified; these are, however, rarely known. The fact that the results depend strongly on those unknown parameters may undermine the credibility of biological conclusions. To properly estimate the weights, high-quality training datasets are needed, in which inferred events are biologically supported.

Many cases of HGT were found by manual inspection of incongruences in gene trees [15]. Inferring a locus tree facilitates such analyses, as it allows to automatically detect the incongruences. This approach has several advantages over reconciliation. It allows to restrict to only two parameters: the locus gain and the locus loss weight. It is also more robust to imprecise data, as improperly placed branches will only be locally detected as new loci, without interfering with the global evolutionary scenario. This allows to disregard the noise when analyzing the tree, and instead focus on several important events. The locus tree inference has been addressed in populational genetics setting [13]. However, this approach requires several difficult to obtain parameters, like speciation times or population sizes.

Our contribution: In this work, we address the problem of Locus Tree Inference when populational effects are negligible. This allows addressing the locus tree inference problem in a parsimony framework. We propose to solve it by decomposing a binary gene tree into a forest of subtrees that can be embedded into a possibly polytomic species tree, in a way that minimizes the weighted sum of the forest size and the number of loss events. We propose two variants of the problem: the *Locus Tree Inference*, *LTI*, in which forest elements are subtrees of the species tree, and the *Conditional Locus Tree Inference*, *CLTI*, where each forest element is a subtree of some binarization of the species tree. We show a dynamic programming algorithm that solves LTI in $O(|G||S|m)$ time and $O(|G||S|)$ space, where m is the maximal degree of a node from the species tree. To solve CLTI, we propose a new mapping, called the Highest Separating Rank. Based on the mapping, we show an $O(d|G| + |S|)$ time and $O(|G| + |S|)$ space algorithm, where d is the height of S, for inferring required and conditional duplications in gene trees, which improves $O(|G|(d + m) + |S|)$ time solution from [18]. Finally, we propose an efficient heuristic to solve CLTI, and present a comparative study on simulated and empirical data.

2 Definitions

Let $T = \langle V_T, E_T \rangle$ be a rooted directed tree. For $a, b \in V_T$, by $\mathrm{lca}_T(a, b)$ we denote the lowest common ancestor of a and b in T. We also use the binary order relation $a \preceq b$ if b is a node on the path between a and the root of T (note that $a \preceq a$). Two nodes a and b are called *siblings* if they are children of $\mathrm{lca}_T(a, b)$. We call a and b *comparable* if $a \preceq b$ or $b \preceq a$, otherwise a and b are called *incomparable*. The parent of a node a is denoted as $\mathsf{parent}(a)$. The subtree of T rooted at v is denoted by $T(v)$. By $L(T)$ we denote the set of all leaves in a tree T and we use $L(v)$ instead of $L(T(v))$. By $\mathrm{root}(T)$ we denote the root of tree T. A

species tree S is a rooted directed tree in which nodes are called *taxa*. A *gene tree* G is a rooted directed binary tree, such that every leaf of G is labeled by a leaf-taxon from S, i.e., an element of $L(S)$. For a node g in G, by $\text{tax}(g) \subset L(S)$ we denote the set of all labels of leaves from $L(g)$.

The *lowest common ancestor mapping*, or lca-mapping, between G and S is a function $M\colon V_G \to V_S$ such that $M(g) = t$ if g is a leaf labelled by the leaf-taxon t, or $M(g) = \text{lca}_S(M(g_1), M(g_2))$ if g is has two children g_1 and g_2. An internal node g in G is a *duplication* if $M(g) = M(g_i)$ for any child g_i of g. Every other node, i.e. a leaf or an internal node satisfying $M(g) \succ M(g_i)$ for every child g_i of g, is called a *speciation* [12, 7, 4].

A node with more than two children is called a *polytomy*. For a polytomy s in a tree S, let $H(s)$ be the set of all possible binary trees whose leaves are the children of s. For instance, if s is the polytomy node present in S from Fig. 2, then $H(s) = \{(d,(e,f)),(e,(d,f)),(f,(d,e))\}$. Let $H^*(S)$ be the set of all possible binary trees obtained from S by replacing each polytomy s with a tree from $H(s)$. An element of $H^*(S)$ is called a *binarization* of S.

3 Locus gain Problems

In this section we introduce the parsimony framework for the (Conditional) Locus Tree Inference problem and a dynamic programming formula for solving the problem. We say that a gene tree G is *embeddable* (respectively, *conditionaly embeddable*) into a species tree S, if each node of G is a speciation (respectively, a speciation in some binarization of S). For instance, $(a,(b,c))$ is embeddable into $(a,(b,c,e),f)$, while $(a,(a,b))$ is not. Since the polytomies in S can be resolved independently, we get the following result:

▶ **Lemma 1.** *G is conditionally embeddable into S if and only if there is a binarization S' of S such that G is embeddable into S'.*

Every internal node g of G induces a set of *loss* events defined as nodes of the species tree strictly between $M(g)$ and $M(\mathsf{parent}(g))$, plus $M(g)$ if $M(g)$ is a polytomy. The above definition yields a notion of the *loss cost*, denoted by $\Lambda(G,S)$, and defined as the total number of loss events required to embed G into S.

A gene tree may not be embeddable into a species tree due to duplications or HGTs. Our goal is to decompose a gene tree into a set of embeddable subtrees in the most parsimonious way. We say that a forest F is a *decomposition* of G, if $\bigcup_{T\in F} L(T) = L(G)$, and for every $T \in F$, $G|_{L(T)} = T$, where, for $A \subseteq L(G)$, $G|_A$ is a tree induced by A and the set of internal nodes $\{\text{lca}_G(a,b) | a,b \in A\}$ and inheriting the ancestor relation from G. Decompositions can be equivalenty obtained by tree edit operations as follows. Given a gene tree G, $X \subseteq E_G$ is called a *set of cuts* if no two edges in X share their top nodes.

▶ **Lemma 2.** *Let X be a set of cuts from G. Let G^X be a graph obtained from G by: removing all cuts from G, contracting all nodes with one parent and one child, and then removing all roots having exactly one child. Then G^X is a decomposition of G.*

In the context of X every node of G can be uniquelly associated with a node from G^X by a mapping σ^X that maps a node g to the first non-removed node below g. Formally, $\sigma^X(g) = \sigma^X(g_1)$ if $\langle g, g_2\rangle \in X$ and g_1 is the sibling of g_2, and $\sigma^X(g) = g$, otherwise. By $M^X\colon G \to S$ we denote the "locus-aware" lca-mapping, given by $M^X(g) = M'(\sigma^X(g))$, where M' is the lca-mapping between $T \in G^X$ and S such that $\sigma^X(g) \in T$ (see Fig. 1).

Consider a set of cuts X in G. We say that X *detaches* $g \in G$, or is *g-detaching*, if $\sigma^X(g)$ is the root of some tree from G^X. For example, in Fig. 1, the cuts from the right example

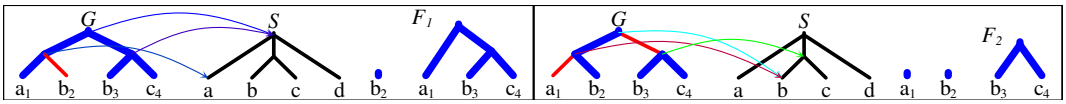

Figure 1 An example of locus trees for $G = ((a_1, b_2), (b_3, c_4))$ with two decompositions F_1 and F_2 consistent with $S = (a, (b, c), d)$. These decompositions are created by cuts indicated with red color. $M^X : G \to S$ is depicted for every set of cuts X (only for internal nodes). Here, $\Lambda(F_1, S) = \Lambda(F_2, S) = 0$, $\Delta(F_1) = 2 \cdot \mathbf{GAIN}$ and $\Delta(F_2) = 3 \cdot \mathbf{GAIN}$, i.e., F_1 is optimal.

detach the parent of leaf a (as $\sigma^X(\mathsf{parent}(a)) = b$ is a root in G^X), while the cuts from the left one do not ($\sigma^X(g) = a$ is below the root).

We say that a decompositon is *consistent* (respectively, *conditionally consistent*) with a species tree S if for every $T \in F$, T is (respectively, conditionally) embeddable into S. From the definition of σ^X we have:

▶ Remark. Let G^X be a decomposition consistent with S. Then, a set of cuts X detaches $g \in G$ if and only if every tree in G^X is either disjoint with or entirely contained in $G(g)$.

Given a species tree S and a gene tree G we define a *locus tree* with respect to S as a pair (G, X), where X is a set of cuts such that the decompositon G^X is consistent with S. Locus trees which induce the same decompositions are considered equivalent. From Lemma 2 it follows that for each set of cuts there is a unique decomposition induced by this set. Conversely, for every decomposition F of G there exists a set of cuts X such that $F = G^X$. Inferring such a set from a given decomposition is straighforward by a bottom-up traversal of the gene tree. Therefore, we can consider decompositions as equivalent to locus trees. From computational point of view, it is more natural to seek for optimal decompositions rather than sets of cuts.

Given a decompositon F, we define the *total loss cost* as $\Lambda(F, S) = \sum_{T \in F} \Lambda(T, S)$. We can now define the *Locus Tree Inference* problem (*LTI*) in the parsimony framework:

▶ **Problem** (Locus Tree Inference, LTI). *Given a gene tree G and a species tree S. Find the decomposition F^* of G consistent with S having the minimal weighted cost $\Delta(G, S) = \mathbf{GAIN} \cdot |F^*| + \mathbf{LOSS} \cdot \Lambda(F^*, S)$ in the set of all decompositions of G consistent with S, where $\mathbf{GAIN} \geq 0$ and $\mathbf{LOSS} \geq 0$ are the weights of locus gain and locus loss events, resp.*

Such decompositions we call *optimal*. In the same way, for conditional consistency, we define the *Conditional Locus Tree Inference* problem (*CLTI*). The problems are equivalent if the input species tree is binary. From the algorithmic point of view, LTI is similar to the reconciliation with DTL scenarios [1] with no duplications. A transfer event corresponds to the creation of a tree in a decomposition forest. Additionally, we do not count loss event at the root of a new tree.

Our algorithm consists of several functions of $g \in G$, $s \in S$ and $\iota \in \{0, 1\}$ which denotes whether a set of cuts detaches g:

D1. $\delta(g, s, 0)$ is the minimal partial cost contribution of $G(g)$ in the set of all g-detaching sets X such that $M^X(g) = s$.

D2. $\delta(g, s, 1)$ as above but for non-g-detaching sets of cuts.

D3. $\delta^{\triangle}(g, s, \iota)$ is the minimal value of $\delta(g, s', \iota)$ for $s' \preceq s$.

D4. $\delta^{\uparrow}(g, s, \iota)$ is the minimal partial cost contribution of $G(g)$ in the set of all g-detaching sets of cuts X such that $M^X(g) \preceq s$. For $\iota = 1$, the cost additionally includes all losses created by $\sigma^X(g)$ and associated with every species node s' satisfying $M^X(g) \prec s' \preceq s$.

Let $c(v)$ be the set of children of v ($\emptyset$ for leaves). By $\mathbf{1}$ we denote the indicator function, that is, $\mathbf{1}[p]$ is 1 if p is satisfied and 0 otherwise. Then, we have the following dynamic programming formula (*DP algorithm*) that solves LTI:

$$\delta(g,s,\iota) = \begin{cases} 0 & \text{if } g \text{ is a leaf and } M(g)=s, \\ \min\{\alpha,\gamma\} & \text{if } g \text{ is not a leaf,} \\ +\infty & \text{otherwise,} \end{cases}$$

where

$$\begin{aligned} \alpha &= \mathbf{1}[c(s)\geq 3]\cdot \mathbf{LOSS}\cdot\iota + \min_{s',s''\in c(s) \text{ and } s'\neq s''} \delta^{\uparrow}(g',s',1)+\delta^{\uparrow}(g'',s'',1), \\ \gamma &= \mathbf{GAIN} + \min(\delta^{\triangle}(g',M(g'),0)+\delta^{\uparrow}(g'',s,\iota), \delta^{\triangle}(g'',M(g''),0)+\delta^{\uparrow}(g',s,\iota)), \\ \delta^{\uparrow}(g,s,\iota) &= \begin{cases} \delta(g,s,\iota) & \text{if } s \text{ is a leaf,} \\ \min\{\delta(g,s,\iota), \mathbf{1}[|c(s)|>1]\cdot\mathbf{LOSS}\cdot\iota + \min_{x\in c(s)}\delta^{\uparrow}(g,x,\iota)\} & \text{otherwise,} \end{cases} \\ \delta^{\triangle}(g,s,\iota) &= \min\{\delta(g,s,\iota), \min_{x\in c(s)}\delta(g,x,\iota)\}. \end{aligned}$$

▶ **Theorem 3.** *For every G and S: $\Delta(G,S) = \min_{s\in S}\delta^{\triangle}(\mathrm{root}(G),s,0)) + \mathbf{GAIN}$.*

Proof. The proof is by induction on the structure of G and S, where the properties D1–D4 of all δ's are proved. We omit technical details. ◀

▶ **Theorem 4.** *The optimal cost can be computed in $O(|G||S|m)$ time and $O(|G||S|)$ space, where m is the maximal degree of a node from S.*

Proof. *Time:* We show that all values of δ functions can be computed in $O(m)$ time. This is straighforward for all values except α, where computing min potentially requires $O(m^2)$ time. This can be done, however, in $O(m)$ time, by finding for each node g' of G, the two children of s with the minimal and the second minimal value of $\delta^{\uparrow}$ and choosing the minimal pair one among all four variants. *Space:* Obvious. ◀

CLTI can be solved by an algorithm similar to the one presented above. It requires additional case in δ for resolving duplications. In order to model a proper binarization of a polytomy in $M(g)$, both children of g have to be mapped into a disjoint sets of children of $M(g)$. Such solution requires extending all δ's by a set of species nodes allowed for the mappings. In consequence, this approach has an exponential time and space complexity. In general we do not know if there is a polynomial time algorithm for CLTI. However, when the locus gain weight (**GAIN**) is much greater than the loss weight (**LOSS**), an efficient heuristic can be constructed, based on a mapping introduced in the next section.

4 Ranked Trees and Rank-based Mappings

Usually, when comparing trees, mappings based on their topologies are used (e.g. the lca-mapping). However, biological species trees contain additional useful structure: the *taxonomic ranks*, like species, genus, or family. Several major ranks are common to almost all living organisms. In this section, we propose a mathematical formalization of ranks and two rank-based mappings, which are useful in duplication inference and CLTI.

A *ranked species tree* is a species tree S in which every node s of S has assigned a small positive integer called *rank*, denoted $R(s)$, such that, for every s and s', if $s \prec s'$ then $R(s) < R(s')$. We assume that the rank of the root of is $d > 0$ and every leaf has rank 1.

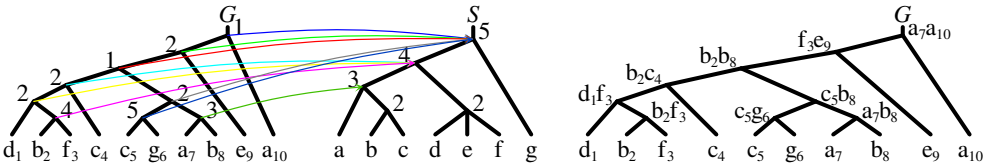

Figure 2 An example of a gene tree G and a species tree S. *Left:* The lca-mapping M. Each internal node of S is decorated with its rank based on the height of the corresponding subtree. Each internal node of G is decorated with the value of mapping P. *Right:* G in which each internal node is decorated by a pair of gene leaves that induces the value of mapping P in Alg. 1 (line 7). For example, for the left child of the root, say x, f_3e_9 yields $P(x) = R(\mathrm{lca}_S(f, e)) = 2$.

Let G be a gene tree and S be a ranked species tree. For a rank r and a leaf t in S, the unique directed path in S consisting of all taxa comparable with t having the rank lower than r will be called *an (evolutionary) r-lineage* of t. Note that every 1-lineage is empty. We say that leaf-taxa t and t' are *separated* by the rank r if for every x from the r-lineage of t and every y from the r-lineage of t', x and y are incomparable. Observe that every pair of leaf-taxa is separated by the rank of 1. Moreover, if r separates t and t' then every rank lower than r also separates t and t'. For example, in Fig. 2 leaf-taxa a and c are separated by ranks 1, 2 and 3, but not by rank 4.

Let g be an internal node in G with children g_1 and g_2. The *highest separating rank* mapping $P\colon V_G \to \{1, 2, \ldots, d\}$ is defined as

$$P(g) = \max\{r\colon r \text{ separates every pair of leaf-taxa from } tax(g_1) \times tax(g_2)\}. \tag{1}$$

The *lowest common rank* mapping $I\colon V_G \to \{1, 2, \ldots, d\}$ is defined as $I(g) = R(M(g))$. We now present some basic properties of both mappings. Simple proofs are omitted for brevity.

▶ **Lemma 5.** *Let $\rho(t, t')$ be the highest rank that separates leaf-taxa t and t' and let g be an internal node of G with two children g_1 and g_2. Then:*

(A) *For every leaf-taxa t and t', $\rho(t, t') = R(\mathrm{lca}_S(t, t'))$.*
(B) $P(g) = \min\{\rho(t, t') : \langle t, t'\rangle \in tax(g_1) \times tax(g_2)\}$.
(C) $I(g) = \max\{R(\mathrm{lca}_S(t, t')) : \langle t, t'\rangle \in tax(g_1) \times tax(g_2)\}$.
(D) $P(g) = \min\{R(\mathrm{lca}_S(t, t')) : \langle t, t'\rangle \in tax(g_1) \times tax(g_2)\}$.
(E) $P(g) = 1$ *if and only if* $\mathrm{tax}(g_1) \cap \mathrm{tax}(g_2) \neq \emptyset$.

Taxonomic ranks have been used earlier for HGT detection [11]. In this work, the authors decorated nodes of the gene tree with the rank of the lowest taxon shared by each descendant leaf, equivalent with the I mapping. A high difference between the rank of a node and the one of its parent was one of the premisses for HGT. To the best of our knowledge, no mapping equivalent with the highest separating rank has been proposed to day.

4.1 Computing mappings

Given a species tree S and a gene tree G, to compute I we can use the classical algorithm for lca-queries, in which, after a linear-time preprocessing, computing lca-queries can be completed in constant time [2]. We conclude that I can be computed in $O(|G| + |S|)$ time.

A naïve algorithm for computing P, based on Lemma 5, requires $O(|G||S|^2)$ time. Here, we propose an $O(d|G| + |S|)$ time solution. For two distinct leaves l_1 and l_2 of G, we write $l_1 <_p l_2$ if l_1 is visited earlier than l_2 in prefix traversal of G. For instance, in Fig. 2 the leaves are linearly ordered starting from the left, i.e., $d_1 <_p b_2 <_p f_3 <_p \ldots$.

Algorithm 1 Computing P

1: **Input:** A ranked tree S with maximal rank d, a gene tree G such that every leaf of G has an attribute map equal to its label, i.e., a leaf from S. A parent of a node is denoted by attr. parent (None for the root).
Output: Values of P stored in attr. P of each internal node of G.
2: **For** s in S.nodes: **Let** s.lastvisited := None # *initialize last visited leaf*
3: **For** g in G.nodes: **If** g is a leaf **Then** v.smap := v.map **Else** g.P := None # *init* P *and* smap
4: Initialize data structure in G for lca-queries $\text{lca}_G(x, y)$ in G.
5: **For** rank in $1, 2, \ldots, d$: **For** v in G.leaves in prefix order such that v.smap.rank = rank:
6: **If** not v.smap.lastvisited = None **Then**
7: $g := \text{lca}_G(v, v.\text{smap.lastvisited})$
8: **If** g.P = None **Then** g.P := rank # P *assignment*
9: v.smap.lastvisited := v # *update last visited*
10: v.smap := v.smap.parent # *climb in* S

▶ **Lemma 6.** *For a fixed $s \in S$, the sequence of all assignment evaluations in line 9 such that v.smap $= s$ induces a sequence of values v, denoted by $v_1, v_2, \ldots, v_k$ such that:* (I) *the assignment s.lastvisited := v_i is executed only when* rank $= s$.rank, (II) $v_1 <_p v_2 <_p \cdots <_p v_k$, *and* (III) $\{v_1, v_2, \ldots, v_k\} = M^{-1}(L(s))$.

Proof. (I) is obvious by the condition in the second loop. By the condition in the inner loop, the order of leaves induced by a sequence of such assignments follows $<_p$. For every gene leaf v, v.smap is initially set to the label of v, i.e., $M(v)$ (see line 3). Thus, if s is a leaf, i.e., s.rank $= 1$, then the assignment in line 9 sets the value of v.lastvisited if and only if the label of v is s. Thus for the leaves, (II) is satisfied. For (III), note that the line 10, ensures that every leaf v is assigned once to s.lastvisited of every node s of a species tree that is present on the path starting from $M(v)$ and terminating in the root. Hence, $M^{-1}(L(s)) \subseteq \{v_1, v_2, \ldots, v_k\}$. The other inclusion follows trivially from the fact that for a leaf v, v.smap is originally set to $M(v)$ and v cannot be assigned to a node incomparable with $M(v)$. ◀

▶ **Lemma 7.** *For every internal node g, $P(g) = g$.P.*

Proof. Let g' and g'' be the left and the right child of g, respectively. The proof is by induction on the rank $r = 1, 2, \ldots, d$. Let $r = 1$. Assume that $P(g) = 1$, we show that g.P $= 1$. Let $s \in \text{tax}(g') \cap \text{tax}(g'')$. Then, by Lemma 6, let Λ_s be the sequence $\{v_1, v_2, \ldots, v_k\}$ of all leaves assigned to s.lastvisited such that $M(v_i) = s$ and ordered by $<_p$. Clearly, the list has the leaves from both subtrees of g, thus there is an index $j < k$, such that $v_j \in L(g')$ and $v_{j+1} \in L(g'')$. Thus $\text{lca}_G(v_j, v_{j+1}) = g$. Now, in line 8, when v is v_{j+1} then v.smap.lastvisited is v_j. In such a case, either g.P is None and g.P will be set to 1, or g.P is already set, however, it can be only 1. This completes the first part of the proof.

Assume that $P(g) = r$ and for every q, such that $P(q) < r$, we have $P(q) = q.P$. For every $v \in L(g')$ and $w \in L(g'')$, $R(\text{lca}_S(M(v), M(w))) \geq r$, thus $g.P =$ None, when Alg. 1 starts the main loop with rank $= r$. From Lemma 5, there is a pair taxa $\langle t_1, t_2 \rangle \in \hat{g}$ such that $s = \text{lca}_S(t, t')$ and $R(s) = r$. Thus, there are two leaves a_1 and a_2 in G such that for each i, $M(a_i) = t_i$ and $\text{lca}(a_1, a_2) = g$, i.e. $a_1 \preceq g'$ and $a_2 \preceq g''$. Similarly, to the first step, the leaves from $M^{-1}(L(s))$ are all visited and set to s.lastvisited according to the order $<_p$. The sequence contains elements a_1 and a_2, therefore again there is j separating leaves from both subtrees of g. The rest of the proof is analogous: in line 8 either $g.P$ is already set to r (if there was other s', processed before s, with $R(s') = r$ satisfying the same properties as s) or it will be set in to r. This completes the proof. ◀

▶ **Lemma 8.** *Alg. 1 requires $O(d|G| + |S|)$ time and $O(|G| + |S|)$ space.*

Proof. *Time:* Lines 2–5 have $O(|G| + |S|)$ time complexity, while the body of the inner loop needs $O(1)$ time. *Space:* Alg. 1 uses only a few node attributes plus the lca-query data structure of the size $O(|G|)$. ◀

5 Classification of Gene Duplications

Several methods for reconciliation with non-binary gene trees have been proposed [20, 16, 19, 6, 3, 9]. However, reconciliation with non-binary species trees is harder to model. This is because a polytomy may represent a lack of knowledge about the order of speciations, and therefore some duplication nodes may correspond to biological speciations. This motivates a further classification of duplication nodes into conditional and required duplications [18].

When reconciling a gene tree G with every binarization of S, if g from G is a duplication in every reconciliation, then g is called a *required duplication*. Similarly, if g is a duplication in at least one but not all reconciliations, we say that g is a *conditional duplication*. Note that G is conditionally embeddable in S if and only if each node in G is either a speciation or a conditional duplication.

In this section, we show how P and I can be used to solve the problem of gene duplication classification when the species tree has possible polytomies.

▶ **Lemma 9.** *For an internal node g from a gene tree G, the following conditions are equivalent:*

(A1) $P(g) = I(g)$,

(A2) *every subtree rooted below $M(g)$ contains taxa from at most one child of g, i.e., for every $s \prec M(g)$, if $L(s) \cap \text{tax}(g_1) \neq \emptyset$ then $L(s) \cap \text{tax}(g_2) = \emptyset$, where g_1 and g_2 are the children of g, and*

(A3) *for every $\langle t, t' \rangle \in tax(g_1) \times tax(g_2)$, $\text{lca}_S(t, t') = M(g)$.*

Proof. *(A1) ⇒ (A2).* Assume that $s \prec M(g)$ and there are two leaves t and t' in $L(s)$ such that $t \in \text{tax}(g_1)$ and $t' \in \text{tax}(g_2)$. Hence, $\langle t, t' \rangle \in tax(g_1) \times tax(g_2)$ and $\text{lca}_S(t, t') \preceq s \prec M(g)$. Thus, $P(g) < I(g)$, a contradiction. *(A2) ⇒ (A3).* Let $\langle t, t' \rangle \in \hat{g}$. Then, t and t' are leaves from two different subtrees rooted below $M(g)$. Therefore, $\text{lca}_S(t, t') = M(g)$. *(A3) ⇒ (A1).* It follows immediately from Lemma 5. ◀

Note that the above Lemma holds also when $P(g) = I(g) = 1$, i.e. when an internal node g is mapped to a leaf of S. In such a case the condition (A2) is satisfied trivially.

▶ **Lemma 10.** *Let $P(g) = I(g)$. Then, g is a speciation iff $M(g)$ is an internal node and there are exactly two subtrees rooted at children of $M(g)$ having nodes from* $\text{tax}(g)$.

Proof. ($\Rightarrow$). We have that g is an internal node. In such a case $I(g) > 1$ and $M(g)$ is an internal node. Then, by (A2) from Lemma 9, every child of $M(g)$ has taxa present in at most one child of g. Clearly, there are at least two children of $M(g)$ satisfying this property. If there are more than two, then one child of g, say g_1, has taxa from at least two children of $M(g)$. Hence, $M(g_1) = M(g)$ and g is a duplication node, a contradiction. ($\Leftarrow$). Similarly, if $M(g)$ is an internal node, then by (A2), the mappings of the children of g are incomparable and located below $M(g)$, therefore g cannot be a duplication. ◀

We have a symmetric property whose proof is similar to the previous one.

▶ **Lemma 11.** *Assume that $P(g) = I(g)$. Then, g is a duplication node if and only if either $M(g)$ is a leaf or $M(g)$ is an internal node and there are at least three subtrees rooted at a child of $M(g)$ having nodes from* $\mathrm{tax}(g)$.

Finally, we have the main property.

▶ **Theorem 12** (Classification Theorem)**.** *Let g be an internal node of G. Then:*
(C1) *If $P(g) = I(g) = 1$ or $P(g) < I(g)$ then g is a required duplication.*
(C2) *If $P(g) = I(g) > 1$, then g is a duplication if and only if g is a conditional duplication.*

Proof. *(C1)* If $P(g) = I(g) = 1$, then g is mapped to a leaf. Hence, every leaf below g has the same label. Thus, in every binarization of S, g is a duplication. Assume that $P(g) < I(g)$. Then $M(g)$ is an internal node in S, having at least three taxa in $L(M(g))$ (otherwise, the two children of $M(g)$ are leaves and $P(g) = I(g) = 2$). We can assume that there are three leaves $t, t' \in \mathrm{tax}(g_1)$ and $t'' \in \mathrm{tax}(g_2)$ such that $\mathrm{lca}_S(t', t'') \prec \mathrm{lca}_S(t, t', t'')$. Clearly, this property holds for every binarization T of S, where the possible polytomy $M(g)$ is resolved. Moreover, in every T, $M(g_1) \succeq \mathrm{lca}_T(t, t', t'')$, thus $M(g_1)$ is comparable with $M(g_2) \succeq t''$. Thus, $M(g) = \max(M(g_1), M(g_2))$ and g is a duplication node.

(C2,⇐). If g is a conditional duplication, then it is a duplication by the definition. *(C2,⇒).* Assume that g is a duplication, then by condition (A2) from Lemma 9, the children of $M(g)$ can be clustered into three disjoint sets X, X' and X'' such that every node from X has no taxa present in $\mathrm{tax}(g)$, every node of X' has taxa from $\mathrm{tax}(g')$ but not from $\mathrm{tax}(g'')$ and analogously every node of X'' has taxa from $\mathrm{tax}(g'')$ but not from $\mathrm{tax}(g')$, where g' and g'' are the children of g. In addition, by Lemma 11 at least one among X' and X'', say X', has at least two elements. Consider a binary tree T in $H(M(g))$, such that all elements of X' and X'' are located in the left and the right subtree of T, respectively. Then, $\mathrm{lca}_T(X')$ and $\mathrm{lca}_T(X'')$ are incomparable. Thus, in such a binarization of S, g' and g'' maps below $M(g)$, and g is a speciation node. Similarly, it can be shown that there exist a tree in $H(M(g))$ in which g is a duplication. ◀

Based on Alg. 1, Classification Theorem leads to a natural $O(d|G| + |S|)$ time solution for the inference of required and conditional duplications when reconciling a given binary gene tree with a species tree. This improves known $O(|G|(d + m) + |S|)$ time algorithm from [18], where m is the maximal degree of a node from S. The improvement is beneficial for highly polytomic species trees. For example, as of 04.28.2017, the genus *Aspergillus* has 1950 children species in the NCBI Taxonomy.

6 Heuristic for CLTI

In this section, we propose the heuristic algorithm for CLTI when the locus gain weight is much greater than the loss weight. The algorithm is based on the following lemma, which follows directly from Theorem 12:

▶ **Lemma 13.** *Tree G is conditionally embeddable in S if and only if for all internal nodes g in G, $I(g) = P(g) > 1$.*

Alg. 2 is a greedy approach that iteratively finds the minimal nodes g such that $P(g) < I(g)$ or $I(g) = 1$ and detaches an embeddable subtree below each node. Note the following:

▶ Remark. Let G be conditionally embeddable in S. Let $\hat{\Lambda}(G, S) = |L(M(\mathrm{root}(G)))| - |L(G)|$. Then, $\Lambda(G, S) \leq \hat{\Lambda}(G, S)$.

Algorithm 2 Heuristic algorithm for CLTI

1: **Input:** A ranked tree S and a gene tree G.
Output: A decomposition forest of G conditionally consistent with S.
Initialize: $F := \emptyset$
2: Compute I and P in G.
3: **Let** Z be the set of all minimal nodes g in G such that $G(g)$ is not conditionally embeddable (i.e. $P(g) < I(g)$ or $I(g) = 1$). **If** Z is empty **Then Return** $F \cup \{G\}$
4: **For** every g in Z:
5: **Let** $\langle v, w\rangle$ be the edge incident to a child of g with minimal $\hat{\Lambda}(G(w), S) + \hat{\Lambda}(G(g) \setminus G(w), S)$ such that $G(w)$ and $G(g) \setminus G(w)$ are conditionally embeddable.
Add $G(w)$ to F, remove e and $G(w)$ from G.
6: Repeat steps 2-6 until tree G is empty.

Figure 3 Comparison of DP and heuristic algorithms for binary species trees in terms of forest size $|F|$ and numbers of losses $\Lambda(F, S)$. The brown line depicts the median cost; the grey ribbon depicts the 90% confidence interval. The weights in the DP algorithm have been set to $\mathbf{GAIN} = 1000, \mathbf{LOSS} = 1$. The plots have been smoothed with cubic splines.

Let $T_1 \setminus T_2$ denote tree T_1 with detached subtree T_2. Then, $\hat{\Lambda}(G', S) + \hat{\Lambda}(G(g) \setminus G', S)$ is an estimate of the partial loss cost induced by detaching subtree G'. The detached subtree in Alg. 2 is chosen so as to minimize this estimate. To limit the complexity of a single step, we consider only subtrees rooted at vertices at a close neighbourgood of g.

A major advantage of Alg. 2 is the space complexity, which is $O(|G| + |S|)$. This makes the heuristic suitable for trees with hundreds or thousands of nodes. The time complexity is $O(ad|G| + a|S|)$, where a is the number of recomputations of I and P mappings. Pessimistically, $a = O(|G|)$, which makes this algorithm assymptotically quadratic. However, in applications this number is expected to be a small integer.

6.1 Experimental validation

In the case of binary species trees, conditional embeddability is identical to strict embeddability, and both locus tree inference algorithms can be compared experimentally.

For each $|L(G)| = 1, \ldots, 20$ and $|L(S)| = 10$ we have generated 100 pairs of random trees under the Yule-Harding model. The leaves of G have been assigned to leaves of S randomly. The numbers of losses for heuristic algorithm have been computed using a modification of the DP algorithm. The inferred costs are shown in Fig. 3.

The costs are similar for both algorithms. The forest size is approximately half the number of leaves in G. Using linear regression, we have determined that, on average, the inferred forest size is equal to $0.47|L(G)|$ for DP and $0.53|L(G)|$ for the heuristic. The number of losses is slightly smaller for the heuristic algorithm (on average $0.98|L(G)|$ for DP and $0.90|L(G)|$ for the heuristic). The reason for this is that the greedy approach tends to detach more concise trees.

7 Example of evolutionary history decomposition

We have compared our approach with a state-of-the-art reconciliation program, Notung 2.9 [17]. We have analyzed the evolution of the gene family of an aminotransferase from a fungus *Penicillium lilacinoechinulatum*[1] (GenBank: *ABV*48733.1), which has been earlier reported to undergo a HGT [15]. The gene tree has been reconciled with NCBI Taxonomy with loss weight 1, duplication weight 1.5 and transfer weight varying from 3 to 8. The result of decomposition by our heuristic approach is depicted in Fig. 4. Depending on the transfer weight, Notung 2.9 reported from 1 to 7 transfers and numerous duplications, while our heuristic inferred a unique result.

Several biological conclusions can be drawn from the decomposition. There are two probable HGTs: into the ancestors of family Clavicipitaceae (blue subtree) and genus *Fusarium* (turquoise subtree). Those groups are distantly related to species in their "mother locus" subtrees, which makes multiple duplications and losses less likely than a HGT. Since both recipient groups are pathogenic (as well as *Aspergillus fumigatus*, in which the gene has been extensively duplicated), we may expect that the protein *ABV*48733.1 plays a role in their pathogenesis. Both transfers are consistent with reconciliation results.

Note that the P mappings of parents of turquoise and blue subtrees are much higher than the ones of their children, consistent with a HGT hypothesis. On the contrary, the P mapping of parent of the green subtree is much lower than the ones of its children, consistent with a duplication. This observation can be a basis for an event scoring system to aid the classification of the events. Other locus gain events are ambiguous, both in the case of history decomposition analysis and the tree reconciliation.

8 Discussion

In this work, we have investigated two new problems for locus tree inference in a parsimony framework. We have proposed and analyzed a new mapping, called the Highest Separating Rank, which has been applied to the problems of duplication classification and locus tree inference. The solution to the duplication classification problem has improved the current one by removing dependence on the maximum node degree in species tree from the time complexity. A prototype implementation is publicly availiable at `https://github.com/mciach/LocusTreeInference`. The presented approach will be used to obtain a manually curated dataset of horizontally transferred genes.

Future outlooks. The influence of potential contradictory binarizations on the decomposition needs to be elucidated. LTI should be generalized for non-binary gene trees, as it would allow to collapse nodes with low support, possibly decreasing the forest size. The P mapping can be applied to obtain efficient solutions to the problem of gene tree rooting and supertree construction. Finally, application of automatic event scoring system should be investigated.

[1] Homologs of the protein sequence have been found in 20 fungal species using BLASTp suite. The sequences have been aligned using MAFFT program and trimmed with TrimAL. The phylogenetic tree has been created using PhyML and rooted by setting *Amanita muscaria* as the outgroup. Nodes of the species tree have been collapsed to represent only the following taxonomic ranks: species, genus, family, order, class, phylum, kingdom.

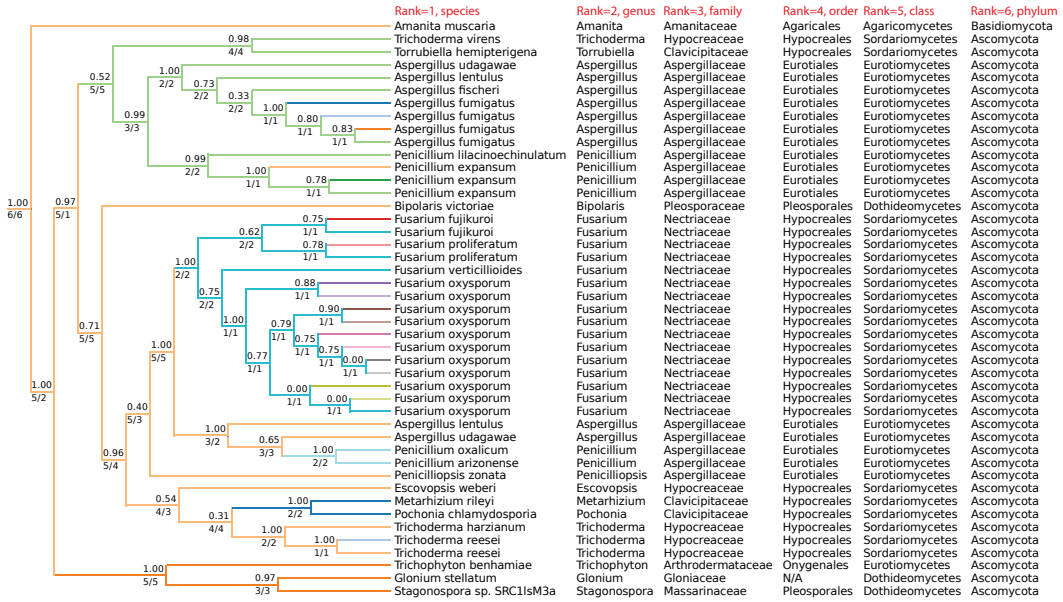

Figure 4 Gene tree of homologs of protein *ABV*48733.1 and evolutionary lineages of organisms. Numbers above branches represent node supports, numbers below branches represent the values of I/P mappings before decomposition. The separate histories of different loci have been highlited by different colors.

References

1 Mukul S. Bansal, Eric J. Alm, and Manolis Kellis. Efficient algorithms for the reconciliation problem with gene duplication, horizontal transfer and loss. *Bioinformatics*, 28(12):i283–i291, 2012.

2 Michael A. Bender and Martin Farach-Colton. The LCA problem revisited. In *Latin American Symposium on Theoretical Informatics*, pages 88–94. Springer, 2000.

3 Ann-Charlotte Berglund-Sonnhammer, Pär Steffansson, Matthew J. Betts, and David A. Liberles. Optimal gene trees from sequences and species trees using a soft interpretation of parsimony. *Journal of Molecular Evolution*, 63(2):240–250, 2006.

4 P. Bonizzoni, G. Della Vedova, and R. Dondi. Reconciling a gene tree to a species tree under the duplication cost model. *Theoretical Computer Science*, 347(1-2):36–53, 2005.

5 Jean-Philippe Doyon, Vincent Ranwez, Vincent Daubin, and Vincent Berry. Models, algorithms and programs for phylogeny reconciliation. *Briefings in Bioinformatics*, 12(5):392, 2011.

6 O. Eulenstein, S. Huzurbazar, and D. A. Liberles. *Evolution after Gene Duplication*, chapter Reconciling Phylogenetic Trees, pages 185–206. John Wiley & Sons, Inc., 2010.

7 P. Górecki and J. Tiuryn. DLS-trees: A model of evolutionary scenarios. *Theoretical Computer Science*, 359(1-3):378–399, 2006.

8 Patrick J. Keeling and Jeffrey D. Palmer. Horizontal gene transfer in eukaryotic evolution. *Nature Reviews Genetics*, 9(8):605–618, 2008.

9 Manuel Lafond, Krister M. Swenson, and Nadia El-Mabrouk. An optimal reconciliation algorithm for gene trees with polytomies. In *International Workshop on Algorithms in Bioinformatics*, pages 106–122. Springer, 2012.

10 Marina Marcet-Houben and Toni Gabaldón. Treeko: a duplication-aware algorithm for the comparison of phylogenetic trees. *Nucleic Acids Research*, page gkr087, 2011.

11 Miguel A Naranjo-Ortíz, Matthias Brock, Sascha Brunke, Bernhard Hube, Marina Marcet-Houben, and Toni Gabaldón. Widespread inter-and intra-domain horizontal gene transfer of d-amino acid metabolism enzymes in eukaryotes. *Frontiers in Microbiology*, 7, 2016.

12 Roderic D. M. Page. Maps between trees and cladistic analysis of historical associations among genes, organisms, and areas. *Systematic Biology*, 43(1):58–77, 1994.

13 Matthew D. Rasmussen and Manolis Kellis. Unified modeling of gene duplication, loss, and coalescence using a locus tree. *Genome Research*, 22(4):755–765, 2012.

14 Matt Ravenhall, Nives Škunca, Florent Lassalle, and Christophe Dessimoz. Inferring horizontal gene transfer. *PLOS Computational Biology*, 11(5):1–16, 05 2015.

15 Thomas A. Richards, Guy Leonard, Darren M. Soanes, and Nicholas J. Talbot. Gene transfer into the fungi. *Fungal Biology Reviews*, 25(2):98–110, 2011.

16 M. J. Sanderson and M. M. McMahon. Inferring angiosperm phylogeny from EST data with widespread gene duplication. *BMC Evolutionary Biology*, 7 (Suppl 1): S3, 2007.

17 Maureen Stolzer, Han Lai, Minli Xu, Deepa Sathaye, Benjamin Vernot, and Dannie Durand. Inferring duplications, losses, transfers and incomplete lineage sorting with nonbinary species trees. *Bioinformatics*, 28(18):i409–i415, 2012.

18 Benjamin Vernot, Maureen Stolzer, Aiton Goldman, and Dannie Durand. Reconciliation with non-binary species trees. *Journal of Computational Biology*, 15(8):981–1006, 2008.

19 Tandy Warnow. Large-scale multiple sequence alignment and phylogeny estimation. In *Models and Algorithms for Genome Evolution*, pages 85–146. Springer, 2013.

20 Yu Zheng and Louxin Zhang. Reconciliation with non-binary gene trees revisited. In *International Conference on Research in Computational Molecular Biology*, pages 418–432. Springer, 2014.