BIOINFORMATICS

Vol. 24 ISMB 2008, pages i132–i138
doi:10.1093/bioinformatics/btn150

The multiple gene duplication problem revisited

Mukul S. Bansal and Oliver Eulenstein*
Department of Computer Science, Iowa State University, Ames, IA 50011, USA

ABSTRACT

Motivation: Deciphering the location of gene duplications and multiple gene duplication episodes on the Tree of Life is fundamental to understanding the way gene families and genomes evolve. The multiple gene duplication problem provides a framework for placing gene duplication events onto nodes of a given species tree, and detecting episodes of multiple gene duplication. One version of the multiple gene duplication problem was defined by Guigó *et al.* in 1996. Several heuristic solutions have since been proposed for this problem, but no exact algorithms were known.

Results: In this article we solve this longstanding open problem by providing the first exact and efficient solution. We also demonstrate the improvement offered by our algorithm over the best heuristic approaches, by applying it to several simulated as well as empirical datasets.

Contact: oeulenst@cs.iastate.edu

1 INTRODUCTION

Gene duplication is known to have played a major role in the evolution of almost all life on Earth. For example, analyses of genomic data from numerous plants such as grasses (Guyot and Keller, 2004; Paterson *et al.*, 2004; Schlueter *et al.*, 2004; Vandepoele *et al.*, 2003; Wang *et al.*, 2005; Yu *et al.*, 2005), *Arabidopsis* or other Brassicaceae (Blanc *et al.*, 2003; Bowers *et al.*, 2003; Cannon *et al.*, 2006; Schlueter *et al.*, 2004; Schranz and Mitchell-Olds, 2006; Simillion *et al.*, 2002; Vision *et al.*, 2000), poplar (Sterck *et al.*, 2005), cotton (Blanc and Wolfe, 2004; Rong *et al.*, 2004) and *Physcomitrella* (Rensing *et al.*, 2007), among others, have revealed evidence of ancient gene duplications. Complex evolutionary processes such as gene duplication and loss, recombination and horizontal gene transfer generate gene trees that differ from species trees. One approach to infer gene duplications is to reconcile the conflicting gene trees with respect to a trusted species tree (Bonizzoni *et al.*, 2005; Chen *et al.*, 2000; Goodman *et al.*, 1979; Górecki and Tiuryn, 2004; Guigó *et al.*, 1996; Mirkin *et al.*, 1995; Page, 1994; Zhang, 1997). Existing techniques that reconcile gene trees to species trees can identify gene duplications but cannot, in general, accurately locate them on the species tree. Other approaches make use of sequence similarity to reconstruct the underlying evolutionary history of genes (see, for example, Wapinski *et al.*, 2007a,b). Probabilistic models for gene/species tree reconciliation as well as gene sequence evolution have also been developed (Arvestad *et al.*, 2003, 2004).

There is evidence that many gene duplications are part of larger *multiple gene duplication episodes*, during which a large portion of an organism's genome is duplicated. In fact, it is known that the entire genomes of numerous species (many eukaryotes for example) have been entirely duplicated one or more times. However, the rapid gene loss and gene rearrangements that follow a multiple gene duplication episode can make them difficult or even impossible to detect; and there is often no clear consensus on the number of ancient multiple gene duplication episodes or their precise location in evolutionary history.

Deciphering the location of gene duplications and multiple gene duplication episodes on the Tree of Life is a fundamental problem in understanding the way gene families and genomes evolve. The multiple gene duplication problem provides a framework for (i) mapping gene duplication events onto a given species tree and (ii) inferring and locating multiple gene duplication episodes. Informally, the *multiple gene duplication problem* is to assign duplication events to nodes in a species trees in such a way that the total number of multiple gene duplication episodes (or simply episodes in short) is minimized. This allows for a 'parsimonious' reconciliation of the gene trees with respect to a trusted species tree, which helps to locate gene duplications, as well as to detect multiple gene duplication episodes, more accurately.

Guigó *et al.* (1996) were the first to address a comprehensive phylogenetic problem that maps duplication events from a collection of rooted, binary gene trees onto a larger rooted binary species tree. They presented a heuristic that could be used to trace back the identified gene duplications to a few multiple gene duplication episodes. Later on, this heuristic approach was refined and restated in more formal terms, and used to study multiple gene duplication episodes in vertebrates by Page and Cotton (2002). Essentially, this heuristic approach sought to solve the multiple gene duplication problem of Guigó *et al.* by solving instead a similar problem which we refer to as the 'episode clustering' problem. An alternative version of the multiple gene duplication problem was introduced by Fellows *et al.* (1998b) which they proved to be intrinsically difficult. Hence, we direct our focus to the work of Guigó *et al.* and Page and Cotton. The episode clustering problem determines duplication events using the Gene Duplication (GD) model from Goodman *et al.* (1979). Each duplication can be placed on any species on a path between the two (not necessarily distinct) most recent species that could have contained the duplication and its parent, respectively. In case the parent does not exist, the path runs between the most recent species for the duplication and the root of the species tree. An example is depicted in Figure 1. The duplications in gene tree G are represented by the three bold vertices. Associated with each bold vertex is its path represented by an interval. For example, the interval $[5,3]$ represents the path $\langle 5,4,3 \rangle$ in the species trees S. Let g denote the node corresponding to the interval $[5,3]$. Species 5 is the most recent species that could have contained g and the parent of species 3, i.e. 2, is the most recent

*To whom correspondence should be addressed.

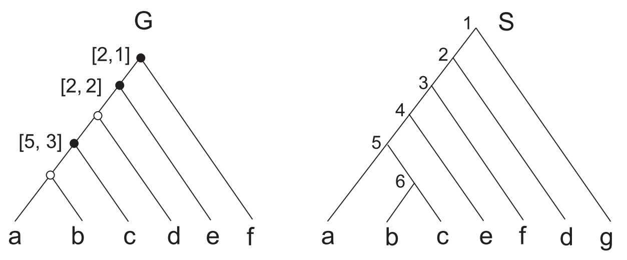

Fig. 1. A gene tree G and a comparable species tree S is depicted. The bold vertices in G are duplications and their intervals represent their allowed locations in the species tree S.

species that could have contained the parent of g. The *Episode Clustering (EC)* problem is, given a collection of gene trees and a species tree, find a minimum number of locations in the species tree where all duplications in the gene trees can be placed. For example, all three duplications in Figure 1 can be placed on species nodes 2 and 3

The EC problem itself has a long and interesting history. Guigó *et al.* (1996) presented a heuristic approach to solve this problem. This heuristic was somewhat imprecise, and there were hints, but no formal algorithm, on how to deal with certain optimization steps. Page and Cotton (2002) observed that the EC problem can be efficiently and cleanly reduced to the Set Cover problem. They approach the EC problem using a heuristic for the intrinsically difficult set-cover problem. Recently, Burleigh *et al.* (2008) gave an efficient and exact solution for the EC problem. However, the EC problem itself suffers from a major limitation: it minimizes the number of locations in the species tree at which gene duplications occur, but it need not minimize the total number of episodes of multiple gene duplication. In fact, it is easy to find examples where minimizing the number of locations, does not minimize the number of episodes. Indeed, the desired goal in the papers of both Guigó *et al.* (1996) and Page and Cotton (2002) is to minimize the number of episodes, and the EC problem was used only as a heuristic approach for this problem. We refer to this problem of minimizing the number of episodes as the *Minimum Episodes (*ME*)* problem. In essence, the ME problem is the multiple gene duplication problem as defined by Guigó *et al.* (1996).

Thus, all previous attempts at solving the ME problem have made use of heuristics approaches based on the EC problem. In this article we finally solve a longstanding open problem by providing the first exact and efficient solution to the ME problem (see Section 3). Our algorithm is surprisingly simple and extremely efficient. We have also implemented our algorithm and demonstrated the improvement it offers over the best heuristic approaches experimentally by applying it to several simulated as well as empirical datasets (see Section 4).

2 BASIC DEFINITIONS, NOTATION AND PRELIMINARIES

In this section we first introduce basic definitions and notation that we shall use and then define the preliminaries required for this work.

2.1 Basic definitions and notation

A *tree* T is a connected graph with no cycles, consisting of a node set $V(T)$ and an edge set $E(T)$. T is *rooted* if it has exactly one distinguished node called the *root* which we denote by $\mathsf{Ro}(T)$. Let T be a rooted tree. We define $\leq_T$ to be the partial order on $V(T)$ where $x \leq_T y$ if y is a node on the path between $\mathsf{Ro}(T)$ and x. The set of minima under $\leq_T$ is denoted by $\mathsf{Le}(T)$ and its elements are called *leaves*. If $\{x,y\} \in E(T)$ and $x \leq_T y$ then we call y the *parent* of x denoted by $\mathsf{Pa}_T(x)$ and we call x a *child* of y. The set of all children of y is denoted by $\mathsf{Ch}_T(y)$. If two nodes in T have the same parent, they are called *siblings*. The *least common ancestor* of a non-empty subset $L \subseteq \mathsf{V}(T)$, denoted as $\mathrm{lca}(L)$, is the unique smallest upper bound of L under $\leq_T$. A *subtree* of T rooted at node $y \in V(T)$, denoted by T_y, is the tree induced by $\{x \in V(T) : x \leq y\}$. T is (fully) *binary* if every node has either zero or two children. Throughout this article, the term tree refers to a rooted fully binary tree.

Given $a \leq_T b$ we define the interval $[a,b] = \{x \in V(T) \mid a \leq_T x \leq_T b\}$. The *height* of T, denoted by $h(T)$ is the number of nodes on a maximal length path from $\mathsf{Ro}(T)$ to a leaf node of T. Thus, a rooted binary tree with three leaves has height three.

2.2 The ME problem

In this section we formally define the ME problem. The ME problem seeks to assign duplication events to nodes in a species tree, where each duplication event is associated with an interval in the species tree describing the locations where that duplication can be placed. The definition of duplication is based on the (GD) model introduced by Goodman *et al.* (1979). Guigó *et al.* (1996) extended this model and defined the associated intervals for each gene duplication. Here we only provide definitions necessary to state the ME problem.

The GD model is based on a gene and species tree from which gene duplications can be derived. A *species tree* is a tree that depicts the evolutionary relationships of a set of species. Given a gene family for a set of species, a *gene tree* is a tree that depicts the evolutionary relationships among the sequences encoding only that gene family in the given species. Thus the vertices in a gene tree represent genes. In order to compare a gene tree G with a species tree S a mapping from each gene $g \in V(G)$ to the most recent species in S that could have contained g is required.

Definition 2.1 (LCA Mapping). *A* leaf-mapping $\mathcal{L}_{G,S}: \mathsf{Le}(G) \to \mathsf{Le}(S)$ *specifies, for each gene g, the species from which it was sampled. The extension* $\mathsf{M}_{G,S}: V(G) \to V(S)$ *of* $\mathcal{L}_{G,S}$ *is the* LCA mapping *defined by* $\mathsf{M}_{G,S}(g) = \mathrm{lca}(\mathcal{L}_{G,S}(\mathsf{Le}(G_g)))$.

Definition 2.2 (Comparability). *The trees G and S are* comparable *if there exists a leaf-mapping* $\mathcal{L}_{G,S}$.[1] *A set of gene trees* $\mathcal{G}$ *and S are* comparable *if each gene tree in* $\mathcal{G}$ *is comparable with S.*

Throughout the remainder of this article, $\mathcal{G}$ denotes a collection of input gene trees, S a comparable species tree, and G denotes an arbitrary gene tree in $\mathcal{G}$.

[1]Note that mathematically speaking such a leaf-mapping always exists. However, in the current context, we are only concerned with biologically relevant leaf-mappings.

DEFINITION 2.3 (Duplication). *A node* $v \in V(G)$ *is a* (gene) duplication *if* $M_{G,S}(v) = M_{G,S}(u)$ *for some* $u \in \mathrm{Ch}(v)$ *and we define* $\mathrm{Dup}(G,S) = \{g \in V(G) \mid g \text{ is a duplication }\}$.

DEFINITION 2.4 (Interval $I(g)$). *For every* $g \in \mathrm{Dup}(G,S)$, *the interval* $I(g)$ *is defined to be:*

- $[M_{G,S}(g), \mathrm{Ro}(S)]$, if $g = \mathrm{Ro}(G)$,
- $[M_{G,S}(g), M_{G,S}(g)]$, if $M_{G,S}(g) = M_{G,S}(\mathrm{Pa}(g))$ and
- $[M_{G,S}(g), M_{G,S}(\mathrm{Pa}(g))] - \{M_{G,S}(\mathrm{Pa}(g))\}$, otherwise.

DEFINITION 2.5 (Valid Mapping). *A mapping* $\mathcal{F}_{G,S} : V(G) \to V(S)$ *is called* valid *if for each* $g \in G$,

$$\mathcal{F}_{G,S}(g) = \begin{cases} s, \text{ where } s \in I(g), & \text{if } g \in \mathrm{Dup}(G,S), \\ M_{G,S}(g), & \text{otherwise.} \end{cases}$$

Note: (i) $\mathcal{F}$ is used to denote the mapping $\cup_{G \in \mathcal{G}} \mathcal{F}_{G,S}$, and we say $\mathcal{F}$ is valid if $\mathcal{F}_{G,S}$ is valid for each $G \in \mathcal{G}$, (ii) given a mapping $\mathcal{F}$, and a node $s \in V(S)$, we write $\mathcal{F}^{-1}(s)$ to denote the set $\{g : \mathcal{F}(g) = s\}$ and (iii) in the remainder of this article, we denote by $\mathcal{F}_M$ the valid mapping $\cup_{G \in \mathcal{G}} M_{G,S}$.

DEFINITION 2.6 ($H(\mathcal{F},s)$). *Given* $\mathcal{G}$ *and* S, *a valid mapping* $\mathcal{F}$, *and a node* $s \in V(S)$, *we define* $H(\mathcal{F},s)$ *to be the sub-graph of* $\mathcal{G}$[2] *induced by the node set* $\mathcal{F}^{-1}(s)$.

Note that $H(\mathcal{F},s)$ must be a forest.

Throughout this article we abbreviate the term ‘multiple gene duplication episode’ simply to ‘episode’. Given any valid mapping $\mathcal{F}$, the following definition defines (i) the number of episodes, $\Delta(\mathcal{F},s)$, at each node $s \in V(S)$ and (ii) the total number of episodes $\Delta(\mathcal{F})$. For the actual definition of an episode itself, we refer the reader to Guigó *et al.* (1996).

DEFINITION 2.7 ($\Delta(\mathcal{F},s)$ and $\Delta(\mathcal{F})$). *Given a valid mapping* $\mathcal{F}$ *and a node* $s \in V(S)$, *we denote by* $\Delta(\mathcal{F},s)$ *the number of episodes at* s *caused by the mapping* $\mathcal{F}$. *Then,* $\Delta(\mathcal{F},s) = \max\{h(T) : T \in H(\mathcal{F},s)\}$, *and,* $\Delta(\mathcal{F}) = \sum_{s \in V(S)} \Delta(\mathcal{F},s)$.

DEFINITION 2.8 (Δ_{opt}). $\Delta_{opt} = \min\{\Delta(\mathcal{F}) : \mathcal{F}$ *is any valid mapping*$\}$.

$\mathcal{G}$ and S form the input for the ME problem. The output is a valid mapping $\mathcal{F}_{opt} : \cup_{G \in \mathcal{G}} V(G) \to V(S)$, such that $\Delta(\mathcal{F}_{opt})$ is minimized. More formally,

PROBLEM 1. *Minimum Episodes (ME)*

Instance:	*A collection of gene trees* $\mathcal{G}$ *and a comparable species tree* S.
Find:	*A valid mapping* $\mathcal{F}_{opt}$ *such that* $\Delta(\mathcal{F}_{opt}) = \Delta_{opt}$.

3 THE ME PROBLEM

It is not hard to see that the number of distinct valid mappings can be extremely large (exponential in the size of the input). It is therefore infeasible to solve the ME problem by considering all possible valid mappings and then picking the one that causes the fewest episodes. In this section we give a simple and extremely efficient algorithm to solve the ME problem. The main idea of the algorithm is to traverse the species tree S in post-order, and perform greedy optimization steps at each node. As we shall prove, this leads us to a globally optimal mapping.

In order to state the algorithm, we must first define a few terms.

DEFINITION 3.1 (Leading node). *Let* $\mathcal{F}$ *be a valid mapping and let* $s \in V(S)$. *Then, we say a node* $g \in \mathcal{F}^{-1}(s)$ *is a* leading node *if and only if* $g = \mathrm{Ro}(T)$ *where* $T \in H(\mathcal{F},s)$ *and* $h(T) = \Delta(\mathcal{F},s)$.

DEFINITION 3.2 (Free node). *Given a valid mapping* $\mathcal{F}$, *and a node* $s \in V(S)$ *such that* $s \neq \mathrm{Ro}(S)$, *a node* $g \in \mathcal{F}^{-1}(s)$ *is called* free *if and only if* $\mathrm{Pa}(s) \in I(g)$.

For convenience, we refer to each node $s \in V(S)$ for which $\mathcal{F}^{-1}(s) \neq \emptyset$, as a *relevant* node. Also recall that $\mathcal{F}_M$ denotes the mapping $\cup_{G \in \mathcal{G}} M_{G,S}$.

We begin by stating the intuitive idea behind our algorithm. Consider any valid mapping $\mathcal{F} : \cup_{G \in \mathcal{G}} V(G) \to V(S)$. Given any node $s \in V(S)$, let $\mathcal{F}'$ be a new mapping constructed from $\mathcal{F}$ by moving the mapping of all the free nodes in $\mathcal{F}^{-1}(s)$ to $\mathrm{Pa}(s)$. Clearly, $\mathcal{F}'$ must be a valid mapping. Now, if all the leading nodes in $\mathcal{F}^{-1}(s)$ are free, then we can show that $\Delta(\mathcal{F}') \leq \Delta(\mathcal{F})$. On the other hand, if not all the leading nodes in $\mathcal{F}^{-1}(s)$ are free, then we must have $\Delta(\mathcal{F}') \geq \Delta(\mathcal{F})$. This simple observation forms the basis of our greedy algorithm. If these greedy optimizations are carried out in a particular order, then it can be shown that the resulting mapping will be an optimal one.

We are now ready to state our algorithm. The algorithm starts with the LCA mapping from the gene trees to the species tree, and progressively modifies it so that when the algorithm terminates, we have an optimal valid mapping. First, a valid mapping $\mathcal{F} : \cup_{G \in \mathcal{G}} V(G) \to V(S)$ is initialized such that $\mathcal{F} = \mathcal{F}_M$. Next, we traverse S in post-order, and at each node, say s, we check if it is relevant and if all the leading nodes in $\mathcal{F}^{-1}(s)$ are free. If they are, then we modify the mapping $\mathcal{F}$ by changing the mapping of all the leading nodes in $\mathcal{F}^{-1}(s)$ to $\mathrm{Pa}(s)$. It can be shown that when the post-order traversal is finished, the mapping $\mathcal{F}$ must be a solution to the ME problem (see Theorem 3.1). This algorithm is described more formally in Algorithm 1.

Algorithm 1 This algorithm solves the ME problem

```
Input: A set of gene trees $\mathcal{G}$ and the species tree $S$. Initially all gene
       nodes map according to the LCA mapping.
1: Compute the LCA mapping $M_{G,S}$ for each $G \in \mathcal{G}$.
2: Compute the interval $I(g)$ for each gene tree node $g$.
3: $\mathcal{F} : \cup_{G \in \mathcal{G}} V(G) \to V(S)$ denotes a valid mapping. Initially, set
   $\mathcal{F} = \cup_{G \in \mathcal{G}} M_{G,S}$.
4: for each node $s$ in a post-order traversal of $S$ do
5:   if $s$ is a relevant node then
6:     if all leading nodes in $\mathcal{F}^{-1}(s)$ are free then
7:       Define a mapping $\widehat{\mathcal{F}} : \cup_{G \in \mathcal{G}} V(G) \to V(S)$ as follows:
           $\widehat{\mathcal{F}}(g) = \begin{cases} \mathrm{Pa}(s), & \text{if } g \text{ is a leading node in } \mathcal{F}^{-1}(s), \\ \mathcal{F}(g), & \text{otherwise.} \end{cases}$   (1)
8:       Rename $\widehat{\mathcal{F}}$ to $\mathcal{F}$
9: return $\mathcal{F}$
```

[2]When $\mathcal{G}$ is viewed as a forest.

We denote the final mapping output by Algorithm 1 by $\mathcal{F}_{opt}$. In the remainder of this section we first show that $\mathcal{F}_{opt}$ is a valid mapping and $\Delta(\mathcal{F}_{opt})=\Delta_{opt}$, and then study the complexity of Algorithm 1.

LEMMA 3.1. *$\mathcal{F}_{opt}$ is a valid mapping.*

PROOF. Algorithm 1 starts with the mapping $\mathcal{F}=\mathcal{F}_{\mathsf{M}}$ which is valid by definition. During each iteration of the loop, the mapping $\mathcal{F}$ may be modified according to Equation (1). However, Equation (1) only modifies the mapping of those nodes that are free, and hence produces a valid mapping. Therefore, each mapping produced by the algorithm, including the mapping $\mathcal{F}_{opt}$, is valid. ■

LEMMA 3.2. *Let $\mathcal{F}\colon \cup_{G\in\mathcal{G}} V(G)\to V(S)$ be any valid mapping. Then we have the following:*

1. *If $s\in V(S)$ is a relevant node under mapping $\mathcal{F}_{\mathsf{M}}$, then $\Delta(\mathcal{F}_{\mathsf{M}},s)-1\leq\Delta(\mathcal{F},s)\leq\Delta(\mathcal{F}_{\mathsf{M}},s)$.*
2. *If $s\in V(S)$ is not a relevant node under mapping $\mathcal{F}_{\mathsf{M}}$, then $0\leq\Delta(\mathcal{F},s)\leq 1$.*

PROOF. *Part 1*: Here s is a relevant node i.e. $\mathcal{F}_{\mathsf{M}}^{-1}(s)\neq\emptyset$. Let A denote the set of nodes that are present in $\mathcal{F}_{\mathsf{M}}^{-1}(s)$ but not in $\mathcal{F}^{-1}(s)$ and B denote the set of nodes that are present in $\mathcal{F}^{-1}(s)$ but not in $\mathcal{F}_{\mathsf{M}}^{-1}(s)$. Observe that all the nodes in A must be leading nodes in $\mathcal{F}_{\mathsf{M}}^{-1}(s)$. Relocating all the leading nodes in $\mathcal{F}_{\mathsf{M}}^{-1}(s)$ reduces $\Delta(\mathcal{F}_{\mathsf{M}},s)$ by exactly 1. Therefore, relocating the nodes in A reduces $\Delta(\mathcal{F}_{\mathsf{M}},s)$ by at most 1. This proves that $\Delta(\mathcal{F}_{\mathsf{M}},s)-1\leq\Delta(\mathcal{F},s)$.

Consider now the set B. Let a and b be two gene duplication nodes from some gene tree in $\mathcal{G}$ such that one is an ancestor of the other, and $\mathcal{F}_{\mathsf{M}}(a)\neq\mathcal{F}_{\mathsf{M}}(b)$. Then, Definition 2.4 implies that $\mathcal{F}(a)\neq\mathcal{F}(b)$. Therefore, none of the nodes in set B is an ancestor of another, and hence none of them is a leading node in $\mathcal{F}^{-1}(s)$. This proves that $\Delta(\mathcal{F},s)\leq\Delta(\mathcal{F}_{\mathsf{M}},s)$.

Part 2: In this case $\mathcal{F}_{\mathsf{M}}^{-1}(s)=\emptyset$, and therefore $B=\mathcal{F}^{-1}(s)$. Following the argument from the previous paragraph, we can conclude that none of the nodes in set B is an ancestor of another. This implies that $\Delta(\mathcal{F},s)\leq 1$. ■

The following three lemmas are required for the proof of Theorem 3.1.

LEMMA 3.3. *Let $\mathcal{F}\colon \cup_{G\in\mathcal{G}} V(G)\to V(S)$ be a valid mapping and $s\in V(S)$ be a node such that $\Delta(\mathcal{F}_{opt},s)>\Delta(\mathcal{F},s)$. Then, $\Delta(\mathcal{F}_{opt},s)=1$.*

PROOF. There are two possible cases: (i) s is not a relevant node under $\mathcal{F}_{\mathsf{M}}$ or (ii) s is a relevant node under $\mathcal{F}_{\mathsf{M}}$. We analyze these cases separately.
Case (i): In this case, by Part 2 of Lemma 3.2, we must have $\Delta(\mathcal{F},s)=0$ and $\Delta(\mathcal{F}_{opt},s)=1$.
Case (ii): If $\Delta(\mathcal{F}_{\mathsf{M}},s)<2$, then by Part 1 of Lemma 3.2 the result follows immediately; therefore, let us assume that $\Delta(\mathcal{F}_{\mathsf{M}},\ s)\geq 2$. Part 1 of Lemma 3.2 implies that we must have $\Delta(\mathcal{F},s)=\Delta(\mathcal{F}_{\mathsf{M}},s)-1$ and $\Delta(\mathcal{F}_{opt},s)=\Delta(\mathcal{F}_{\mathsf{M}},s)$.

Let A denote the set of nodes that are present in $\mathcal{F}_{\mathsf{M}}^{-1}(s)$ but not in $\mathcal{F}^{-1}(s)$, and B denote the set of nodes that are present in $\mathcal{F}_{opt}^{-1}(s)$ but not in $\mathcal{F}_{\mathsf{M}}^{-1}(s)$. All the nodes in A must be leading nodes in $\mathcal{F}_{\mathsf{M}}^{-1}(s)$, and since these nodes are not present in $\mathcal{F}^{-1}(s)$, all the nodes in A must be free as well. Also, none of the nodes in B can be a leading node in $\mathcal{F}_{opt}^{-1}(s)$ (see the proof of Part 1 of Lemma 3.2). Therefore, all of the leading nodes in $\mathcal{F}_{opt}^{-1}(s)$ must be present in A, which implies that all the leading nodes in $\mathcal{F}_{opt}^{-1}(s)$ are free. Thus, during the execution of Algorithm 1, the mapping for these nodes would have been changed. This is a contradiction, and hence we cannot have $\Delta(\mathcal{F}_{\mathsf{M}},s)\geq 2$. ■

LEMMA 3.4. *Let node a be such that $\mathcal{F}_{\mathsf{M}}(a)<_S\mathcal{F}_{opt}(a)$. If $\mathcal{F}\colon \cup_{G\in\mathcal{G}} V(G)\to V(S)$ is a valid mapping such that $\mathcal{F}(a)=\mathcal{F}_{\mathsf{M}}(a)$, then $\Delta(\mathcal{F}_{opt},\mathcal{F}_{\mathsf{M}}(a))<\Delta(\mathcal{F},\mathcal{F}_{\mathsf{M}}(a))$.*

PROOF. Since $\mathcal{F}(a)<_S\mathcal{F}_{opt}(a)$, a must be a leading node in $\mathcal{F}_{\mathsf{M}}^{-1}(a)$. This implies that $\Delta(\mathcal{F},\mathcal{F}_{\mathsf{M}}(a))\nless\Delta(\mathcal{F}_{\mathsf{M}},\mathcal{F}_{\mathsf{M}}(a))$. Moreover, since $\mathcal{F}_{opt}(a)\neq\mathcal{F}_{\mathsf{M}}(a)$, we have $\Delta(\mathcal{F}_{opt},\mathcal{F}_{\mathsf{M}}(a))=\Delta(\mathcal{F}_{\mathsf{M}},\mathcal{F}_{\mathsf{M}}(a))-1$ (see Algorithm 1). The lemma follows. ■

LEMMA 3.5. *Let node a be such that $\mathcal{F}_{\mathsf{M}}(a)<_S\mathcal{F}_{opt}(a)$. If $\Gamma=\{x\colon \mathcal{F}_{\mathsf{M}}(a)<_S x<_S\mathcal{F}_{opt}(a)\}$, then $\mathcal{F}_{opt}(x)=0$ for all $x\in\Gamma$.*

PROOF. Consider any node $x\in\Gamma$. There must exist some valid mapping $\mathcal{F}$, realized during the execution of Algorithm 1, for which $\mathcal{F}(a)=x$. However, as the execution of Algorithm 1 progresses, the mapping of a changes. This implies that a must be a leading node in $\mathcal{F}^{-1}(a)$. Observe that a could be a leading node in $\mathcal{F}^{-1}(a)$ only if $\Delta(\mathcal{F},x)=1$. Furthermore, for the mapping of a to be changed, all the nodes in $\mathcal{F}^{-1}(a)$ must be free, and would therefore not map to node x when the algorithm terminates. Thus, $\mathcal{F}_{opt}(x)=0$. ■

THEOREM 3.1. *Algorithm 1 solves the* ME *problem.*

PROOF. In Lemma 3.1 we have already established that $\mathcal{F}_{opt}$ is a valid mapping. Therefore, to establish the correctness of our algorithm, it is sufficient to show that $\Delta(\mathcal{F}_{opt})=\Delta_{opt}$. Let us assume, for the sake of contradiction, that there exists some valid mapping $\mathcal{F}$ for which $\Delta(\mathcal{F}_{opt})>\Delta(\mathcal{F})$. This implies that there must be at least one node $s\in V(S)$ for which $\Delta(\mathcal{F}_{opt},s)>\Delta(\mathcal{F},s)$. We may assume, without any loss of generality, that s has the following property: there does not exist any other node $t\in V(S_s)$ for which $\Delta(\mathcal{F}_{opt},t)>\Delta(\mathcal{F},t)$.

By Lemma 3.3 we know that node s must be such that $\Delta(\mathcal{F}_{opt},\ s)=1$. This implies that $\Delta(\mathcal{F},\ s)=0$, i.e. $\mathcal{F}^{-1}(s)=\emptyset$. We will now show that there exists at least one node $t\in V(S_s)\setminus\{s\}$ for which $\Delta(\mathcal{F}_{opt},t)<\Delta(\mathcal{F},t)$. This would imply that $\sum_{v\in V(S_s)}\Delta(\mathcal{F}_{opt},v)\leq\sum_{v\in V(S_s)}\Delta(\mathcal{F},v)$.

Let $A=\mathcal{F}_{opt}^{-1}(s)$; clearly $A\neq\emptyset$. We now have two possible scenarios, exactly one of which must be true: (i) $\mathcal{F}(g)>_S s$ for each $g\in A$ or (ii) there exists some $g\in A$ for which $\mathcal{F}(g)<_S s$. If case (i) were possible, it would imply that all nodes in A are leading and free, and therefore Algorithm 1 would have already moved their mappings to nodes that are proper ancestors of s. Hence, case (ii) is the only possible scenario.

So far we have shown that if there exists some valid mapping $\mathcal{F}$ for which $\Delta(\mathcal{F}_{opt})>\Delta(\mathcal{F})$, then there must exist some node, say a, where $a\in A$ and $\mathcal{F}(a)<_S s$. Clearly, $\mathcal{F}_{\mathsf{M}}(a)\leq_S\mathcal{F}(a)\ <_S\ s$. This leads us to two possible cases, exactly one of which must be true: (i) $\mathcal{F}_{\mathcal{M}}(a) = \mathcal{F}(a)$ or (ii) $\mathcal{F}_{\mathcal{M}}(a) <_S \mathcal{F}(a)$. If case (i) were true, then by Lemma 3.4 we must have $\Delta(\mathcal{F}, \mathcal{F}(a)) > \Delta(\mathcal{F}_{opt}, \mathcal{F}(a))$. If case (ii) were true, then Lemma 3.5 implies that $\Delta(\mathcal{F}_{opt}, \mathcal{F}(a)) = 0$, but $\Delta(\mathcal{F}, \mathcal{F}(a)) \neq 0$. Thus, in either case, there exists some $t \in V(S_s) \setminus \{s\}$, for which $\Delta(\mathcal{F}_{opt}, t) < \Delta(\mathcal{F}, t)$. And hence, $\sum_{v \in V(S_s)} \Delta(\mathcal{F}_{opt}, v) \leq \sum_{v \in V(S_s)} \Delta(\mathcal{F}, v)$.

Now, let $P = \sum_{v \in V(S_s)} \mathcal{F}_{opt}^{-1}(v)$ and $Q = \sum_{v \in V(S_s)} \mathcal{F}^{-1}(v)$. Suppose there exists a node x such that $\mathcal{F}_{opt}(x) \in V(S) \setminus V(S_s)$, and $\mathcal{F}(x) \in V(S_s)$. Then, there are two possibilities: (i) $\mathcal{F}(x) \in V(S_s) \setminus \{s\}$ or (ii) $\mathcal{F}(x) = s$. In case (i), Lemma 3.5 implies that $\mathcal{F}_{opt}^{-1}(s)$ must be empty, which is clearly a contradiction. Similarly, case (ii) leads to a clear contradiction as well since $\mathcal{F}^{-1}(s) = \emptyset$. Therefore, such a node x cannot exist. And hence $Q \subseteq P$.

All together, this implies that in the subtree S_s, the mapping $\mathcal{F}$ induces at least as many episodes as the mapping $\mathcal{F}_{opt}$, even though $\sum_{v \in V(S_s)} \mathcal{F}^{-1}(s) \subseteq \sum_{v \in V(S_s)} \mathcal{F}_{opt}^{-1}(s)$. Let us now construct a new valid mapping $\mathcal{F}' \colon \cup_{G \in \mathcal{G}} V(G) \to V(S)$ as follows:

$$\mathcal{F}'(g) = \begin{cases} \mathcal{F}_{opt}(g), & \text{if } \mathcal{F}_{opt}(g) \in V(S_s), \\ \mathcal{F}(g), & \text{otherwise.} \end{cases}$$

In light of the observation made in the previous paragraph, we must have $\Delta(\mathcal{F}') \leq \Delta(\mathcal{F})$. Moreover, $\mathcal{F}'$ has fewer nodes s for which $\Delta(\mathcal{F}_{opt}, s) > \Delta(\mathcal{F}', s)$. Therefore, we can now set $\mathcal{F}$ to be $\mathcal{F}'$ and a straightforward induction argument completes our proof. ■

We now study the complexity of Algorithm 1. In order to simplify our analysis we assume that all $G \in \mathcal{G}$ have approximately the same size. The input for the ME problem is the set of gene trees $\mathcal{G}$, and species tree S. Let $n = |\text{Le}(S)|$, $k = |\mathcal{G}|$ and $m = |\text{Le}(G)|$ for some $G \in \mathcal{G}$.

THEOREM 3.2. *The time complexity of Algorithm 1 is $O(kmn)$.*

PROOF. Computing the LCA mapping for all the gene trees takes $O(kmn)$ time (Zhang 1997). With-in this time, the inverse LCA mapping is also easily computed. All the intervals $I(g)$ can be computed in $O(km)$ time. Now, at each node, s, in the species tree, we must (i) find all the leading nodes in $\mathcal{F}^{-1}(s)$, (ii) check if these leading nodes are free and (iii) update the mapping $\mathcal{F}$. Let $x = |\mathcal{F}^{-1}(s)|$, then, each of the Steps (i), (ii) and (iii) above can be completed in $O(x)$ time. Hence, since $x = O(km)$ and there are $O(n)$ nodes in the species tree, we obtain a total time complexity of $O(kmn)$ for Algorithm 1. ■

4 EXPERIMENTAL RESULTS

We evaluated the efficacy and efficiency of our novel algorithm for the ME problem through comparative studies on simulated and empirical datasets. For this evaluation we implemented our algorithm in the program EXACTMGD. We compared our program against the program APPROXMGD of Burleigh *et al.* (2008) that implements the currently best known approach to solve the ME problem.

Recall that the objective of the ME problem is to produce a mapping which defines the fewest number of episodes. The smaller the number of episodes, the more accurate the mapping. Therefore, for each dataset we compared the programs by measuring three values: (i) The number of episodes defined by the initial LCA mapping (i.e. the unoptimized value), (ii) the number of episodes defined by the mapping produced by APPROXMGD and (iii) the number of episodes defined by the mapping produced by EXACTMGD. All analyses were performed on a 3 Ghz Intel Pentium 4 CPU based PC with Windows XP operating system. A run of EXACTMGD on each of these datasets terminated in less than 1s!

Table 1. Performance of EXACTMGD on simulated datasets

Dataset	Unoptimized	ApproxMGD	ExactMGD
50 taxa	30	28	25
100 taxa	47	38	35
200 taxa	64	54	49
400 taxa	65	45	40

Table 2. Performance of EXACTMGD on empirical datasets

Dataset	Unoptimized	ApproxMGD	ExactMGD
Guigó *et al.* (1996)	9	7	5
Burleigh *et al.* (2008)	1180	1152	1042

4.1 Simulated datasets

Our simulated datasets consist of 50 randomly generated gene trees, all with the same set of taxa, along with a randomly generated species tree.[3] We generated four such datasets, each with a different number of taxa (50, 100, 200 and 400) in the input trees. EXACTMGD shows a significant reduction in the number of episodes as compared to APPROXMGD for each of the four datasets (Table 1).

4.2 Empirical datasets

For the empirical study we evaluated two datasets from the literature. The first dataset consists of 53 gene trees, each representing the evolutionary history of different gene families from a set of 16 eukaryotes. This set was assembled and evaluated for episodes by Guigó *et al.* (1996). Subsequently, this dataset was reused and its evaluation refined by Page and Cotton (2002). The second dataset, assembled and evaluated for episodes by Burleigh *et al.* (2008), consists of 85 gene trees from the Phytome comparative plant genome database and contains genes from 136 plant taxa.

For brevity we refrain from performing biological analyses of our results; and only demonstrate the exceptional level of improvement offered by our algorithm over the best current methods. The results depicted in Table 2 show that the mappings produced by our

[3]Our randomly generated trees have a random (binary) topology and a random assignment of leaf labels.

algorithm are significantly more optimal compared to the mappings produced by the best current approaches. This leads to more accurate inference of the location of gene and genome duplications on the Tree of Life.

5 OUTLOOK AND CONCLUSION

In this article we have provided the first exact and efficient algorithm for a longstanding open problem. Traditionally, the multiple episodes problem has been used to infer the location of episodes of multiple gene duplication on a given species tree. Our new algorithm allows this to be done far more accurately. But another interesting and important fallout of our algorithm is that it may also allow us to infer the 'correct' species trees. Consider the problem of constructing a species tree from conflicting gene trees based on the gene duplication optimality criteria. The *gene duplication problem* is to find for a given set of gene trees a corresponding species tree with the minimum reconciliation cost (Fellows *et al.*, 1998a; Hallett and Lagergren, 2000; Ma *et al.*, 2000; Stege, 1999). However, since these gene duplication events are usually a part of larger multiple gene duplication episodes, it might be more helpful if we can infer species trees directly based on the multiple gene duplication optimality criteria (see also Fellows *et al.* 1998b). Our algorithm offers the first practical and effective means to do so. The idea is to use a local search based hill climbing heuristic to traverse through the search space of possible species tree. Our algorithm can be used to compute the number of episodes induced by each candidate species tree during the local search steps. The lower the number of episodes, the better the species tree.

In addition, it would be interesting to extend the multiple GD model of Guigó *et al.* (1996) by relaxing the constraints on the possible locations of gene duplications on the species tree.

ACKNOWLEDGEMENTS

The authors wish to thank the anonymous referees for their invaluable comments.

Funding: This work was supported in part by National Science Foundation AToL grant EF-0334832.

Conflict of Interest: none declared.

REFERENCES

Arvestad,L. *et al*. (2003) Bayesian gene/species tree reconciliation and orthology analysis using mcmc. In *Proceedings of the Eleventh International Conference on Intelligent Systems for Molecular Biology (ISMB)*. Brisbane, Australia, pp. 7–15.

Arvestad,L. *et al.* (2004) Gene tree reconstruction and orthology analysis based on an integrated model for duplications and sequence evolution. In Bourne,P.E. and Gusfield,D. (eds) *Proceedings of the Eighth Annual International Conference on Computational Molecular Biology (RECOMB)*. ACM, San Diego, California, USA, pp. 326–335.

Blanc,G. and Wolfe,K.H. (2004) Widespread paleopolyploidy in model plant species inferred from age distributions of duplicate genes. *Plant Cell*, **16**, 1093–1101.

Blanc,G. *et al.* (2003) A recent polyploidy superimposed on older large-scale duplications in the *Arabidopsis* genome. *Genome Res.*, **13**, 137–144.

Bonizzoni,P. *et al.* (2005) Reconciling a gene tree to a species tree under the duplication cost model. *Theor. Comput. Sci.*, **347**, 36–53.

Bowers,J. *et al.* (2003) Unravelling angiosperm genome evolution by phylogenetic analysis of chromosomal duplication events. *Nature*, **422**, 433–438.

Burleigh,J.G. *et al.* (2008) Locating multiple gene duplications through reconciled trees. In Vingron,M. and Wong,L. (eds) *Proceedings of the 12th Annual International Conference in Research in Computational Molecular Biology (RECOMB)*. Vol. 4955 in *Lecture Notes in Computer Science*, Springer, Singapore, pp. 273–284.

Cannon,S. *et al.* (2006) Legume genome evolution viewed through the Medicago truncatula and Lotus japonicus genomes. *Proc. Natl Acad. Sci.*, **103**, 14959–14964.

Chen,K. *et al.* (2000) Notung: a program for dating gene duplications and optimizing gene family trees. *J. Comput. Biol.*, **7**, 429–447.

Fellows,M. *et al.* (1998a) Analogs & duals of the MAST problem for sequences & trees. In Chwa,K.-Y. and Ibarra,O.H. (eds) *Proceedings of the 9th International Symposium on Algorithms and Computation (ISAAC'98)*. Vol. 1533 in *Lecture Notes in Computer Science*, Springer, Taejon, Korea, pp. 103–114.

Fellows,M. *et al.* (1998b) On the multiple gene duplication problem. In *9th International Symposium on Algorithms and Computation (ISAAC'98), LNCS 1533*. Springer, Taejon, Korea, pp. 347–356.

Goodman,M. *et al.* (1979) Fitting the gene lineage into its species lineage. a parsimony strategy illustrated by cladograms constructed from globin sequences. *Syst. Zool.*, **28**, 132–163.

Górecki,P. and Tiuryn,J. (2004) On the structure of reconciliations. In Lagergren,J. (ed.) *Recomb Comparative Genomics Workshop 2004*. Vol. 3388, Springer, Bertinoro, Italy.

Guigó,R. *et al.* (1996) Reconstruction of ancient molecular phylogeny. *Mol. Phylogenet. Evol.*, **6**, 189–213.

Guyot,R. and Keller,B. (2004) Ancestral genome duplication in rice. *Genome*, **47**, 610–614.

Hallett,M.T. and Lagergren,J. (2000) New algorithms for the duplication-loss model. In Shamir,R. *et al*. (eds) *Proceedings of the Fourth Annual International Conference on Computational Molecular Biology (RECOMB 2000)*. ACM, Tokyo, Japan, pp. 138–146.

Ma,B. *et al.* (2000) From gene trees to species trees. *SIAM J. Comput.*, **30**, 729–752.

Mirkin,B. *et al.* (1995) A biologically consistent model for comparing molecular phylogenies. *J. Comput. Biol.*, **2**, 493–507.

Page,R.D.M. (1994) Maps between trees and cladistic analysis of historical associations among genes, organisms and areas. *Syst. Biol.*, **43**, 58–77.

Page,R.D.M. and Cotton,J.A. (2002) Vertebrate phylogenomics: reconciled trees and gene duplications. In Altman,R.B. *et al.* (eds) *Proceedings of the 7th Pacific Symposium on Biocomputing (PSB'02)*. Lihue, Hawaii, USA, pp. 536–547.

Paterson,A.H. *et al.* (2004) Ancient polyploidization predating divergence of the cereals, and its consequences for comparative genomics. *Proc. Natl. Acad. Sci.*, **101**, 9903–9908.

Rensing,S. *et al.* (2007) An ancient genome duplication contributed to the abundance of metabolic genes in the moss physcomitrella patens. *BMC Evol. Biol.*, **7**, 130.

Rong,J. *et al.* (2004) A 3347-locus genetic recombination map of sequence-tagged sites reveals features of genome organization, transmission and evolution of cotton (Gossypium). *Genetics*, **166**, 389–417.

Schlueter,J. *et al.* (2004) Mining EST databases to resolve evolutionary events in major crop species. *Genome*, **47**, 868–876.

Schranz,M. and Mitchell-Olds,T. (2006) Independent ancient polyploidy events in sister families Brassicaceae and Cleomaceae. *Plant Cell*, **18**, 1152–1165.

Simillion,C. *et al.* (2002) The hidden duplication past of *Arabidopsis thaliana*. *Proc. Natl. Acad. Sci.*, **99**, 13627–1632.

Stege,U. (1999) Gene trees and species trees: the gene-duplication problem is fixed-parameter tractable. In *Proceedings of the 6th International Workshop on Algorithms and Data Structures, LNCS 1663*, Springer, Vancouver, Canada.

Sterck,L. *et al.* (2005) EST data suggest that poplar is an ancient polyploidy. *New Phytol.*, **167**, 165–170.

Vandepoele,K. *et al.* (2003) Evidence that rice and other cereals are ancient aneuploids. *Plant Cell*, **15**, 2192–2202.

Vision,T. *et al.* (2000) The origins of genome duplications in *Arabidopsis*. *Science*, **290**, 2114–2117.

Wang,X. *et al.* (2005) Duplication and DNA segmental loss in the rice genome: implications for diploidization. *New Phytol.*, **165**, 937–946.

Wapinski,I. *et al.* (2007a) Automatic genome-wide reconstruction of phylogenetic gene trees. In *Proceedings of 15th International Conference on Intelligent Systems for Molecular Biology (ISMB) & 6th European Conference on Computational Biology (ECCB), ISMB/ECCB (Supplement of Bioinformatics)*. Vienna, Austria, pp. 549–558.

Wapinski,I. *et al.* (2007b) Natural history and evolutionary principles of gene duplication in fungi. *Nature*, **449**, 54–61.

Yu,J. *et al.* (2005) The genomes of Oryza sativa: a history of duplication. *PLoS Biol.*, **3**, 266–281.

Zhang,L. (1997) On a Mirkin-Muchnik-Smith conjecture for comparing molecular phylogenies. *J. Comput. Biol.*, **4**, 177–187.