Exact Solutions for Classic Gene Tree Parsimony Problems

Wen-Chieh Chang
Dept. of Computer Science
Iowa State University
Ames, Iowa 50011, USA
wcchang@iastate.edu

André Wehe
Dept. of Biology
University of Florida
Gainesville, Florida 32611, USA
awehe@ufl.edu

Paweł Górecki
Dept. of Mathematics, Informatics
and Mechanics
University of Warsaw
Banacha 2, 02-097 Warsaw, Poland
gorecki@mimuw.edu.pl

Oliver Eulenstein
Dept. of Computer Science
Iowa State University
Ames, Iowa 50011, USA
oeulenst@iastate.edu

Abstract

Gene tree parsimony (GTP) problems seek a species tree that infers the fewest evolutionary events necessary to explain discordance in a given collection of gene trees. Solving GTP problems allows evolutionary analyses of large gene families with complex evolutionary histories, for example, where gene duplication and loss events, or deep coalescence (incomplete lineage sorting), play a major role. Such problems are NP-hard, and therefore are addressed by heuristics, which lack any performance guarantee. We describe fast dynamic programming algorithms using bit-vector representation and Gray code enumeration that solve the GTP problems exactly, and in addition report the number of all optimal solutions. We show that our algorithms can solve instances with data sets consisting of as many as 22 taxa. Finally, we demonstrate the performance of our exact algorithms using empirical and simulated studies, and compare our exact results with those of GTP heursistics.

1 Introduction

Modern sequencing technologies have provided deep and rich data sets of genetic sequences for inferring gene phylogenies (gene trees) from which organismal phylogenies (species trees) can be estimated. Such estimations are based on the basic assumption that evolutionary relationships in gene trees provide information about phylogenetic relationships of the species from which the genes have been isolated [9]. Evolutionary studies often assume that correctly inferred gene trees represent the species trees of the species sampled. However, due to evolutionary events like gene duplication, gene loss, and deep coalescence, there are often discrepencies between correctly inferred gene trees and the species tree within which the genes evolved [9, 12, 24]. A classic approach to infer species trees from a collection of discordant gene trees is to utilize gene tree parsimony (GTP) problems. These problems seek a species tree that minimizes the total number of evolutionary events required to reconcile the discordance in a given collection of gene trees [24].

GTP problems have been well studied in the literature and shown to be NP-hard for evolutionary events of duplication, duplication and loss, and deep coalescence [17, 28]. Other negative time complexity results have been shown for GTP problems [3, 4]. In practice, this intractability has been addressed by using local search heuristics [7, 19, 26]. While these heuristics have found some reasonable species tree estimates [9, 21], they lack provable solution qualities and runtime bounds. Therefore, it may be desirable to pursue exact solutions for GTP problems.

Every NP-hard problem can be solved by exhaustive enumeration of all candidate solutions. However, the run time is proportional to the number of candidate solutions, and typically becomes prohibitively large even for small instances. To overcome this limitation, algorithms that are substantially faster than exhaustive enumeration have been developed for many NP-hard problems [5, 8, 15]. While only few exact solutions have been developed for GTP problems [6, 23, 25], they have already provided new insights into the biological accuracy of GTP problems [23], performance evaluations of GTP heuristics [25], and the first exact species trees for smaller phylogenetic studies based on GTP [6, 25]. For example, a study of exact solutions showed that GTP correctly inferred the trusted species tree with strong statistical support [23], and it was

demonstrated that an exact algorithm was able to compute a species tree of a major seed plant lineage with 12 taxa using 6,084 gene trees [6].

We introduce exact dynamic programming algorithms for each of the described GTP problems (duplication, duplication and loss, and deep coalescence). Our algorithms are optimized by using minimized space consumption, bit-vector representation, and Gray encoding for sub-problem enumeration. In addition, our algorithms also provide the number of all optimal species trees. Finally, we demonstrate that our algorithms can compute instances with up to 22 taxa for gene duplications, gene duplication and loss events, and deep coalescences. The implementation of our algorithm is freely available from the authors.

Related Work. Only a few GTP problems have been addressed by exact solutions. The GTP problem for gene duplications could be solved exactly for instances with up to 8 taxa by exhaustive enumeration [23], and up to 14 taxa using an integer linear programming (ILP) formulation [6]. The GTP problem for deep coalescence events has been solved exactly for instances with up to 8 taxa using dynamic programming and an ILP formulation [25].

In order to reduce the search space of species trees, a variation of GTP problems was proposed where consistent knowledge of the resulting species tree was provided as part of the input [11]. Similarly, knowledge-enhanced GTP problems add to the original instance a set of inconsistent knowledge, and seek an optimal tree within all trees that can be built from parts of the input knowledge. Exact algorithms have been described for such GTP problems for several evolutionary events. These algorithms are based on dynamic programming and have a polynomial run time in the size of the input gene trees and the knowledge [27].

Our Contribution. We introduce dynamic programming algorithms that use bit-vector representation and Gray encoding to solve the GTP problems exactly. This allows us to efficiently utilize large registers of modern processors. As a result, our algorithms have complexities in $\Theta(3^n mn/b)$ time and $\Theta(2^n + m)$ space, where n is the overall number of taxa presented in input trees, m is the number of unique rooted splits in the input gene trees, and b is the size of the bit-vector used. To the best of our knowledge, the largest species trees described in the literature that were exactly computed for non-trivial GTP instances had 8 taxa for the deep coalescence cost function and 14 taxa for the gene duplication events. Using simulation studies we demonstrate that our algorithm can compute species trees with up to 22 taxa for each of the GTP problems in less than 17 hours on a standard workstation. Using our exact algorithms we showed that species trees inferred by the GTP heuristic implemented in DupTree [26] are largely misled when using instances with at least 16 taxa. Our empirical study with an instance consisting of gene family trees from 16 taxa showed that DupTree reported optimal species trees in only 8% of one thousand runs. This makes our exact algorithms a viable alternative to DupTree for instances with smaller taxa numbers. Further, our algorithms also report the number of all optimal solutions, which can be beneficial in phylogenetic analyses. For example, our experiments confirmed, for the first time, that all optimal solutions in a published empirical study have already been identified.

2 Preliminaries

A *gene tree* is a rooted binary tree (DAG) whose leaves are labeled by species names. Let T be a gene tree. By $L(T)$ we denote the set of all leaf labels present in T. A node n is called *internal* if it has two children, which are denoted by $Ch(n)$. A cluster of a node g is the set of leaf labels visible from g. A *species tree* is a gene tree whose leaves are uniquely labeled.

Let $S = \langle V_S, E_S \rangle$ be a species tree. For nodes $a, b \in V_S$, let $a + b$ be the least common ancestor of a and b in S. We also use the binary order relation $a \leq b$ if b is a node on the path between a and the root of S. Two nodes a and b are called *siblings* if they are children of $a + b$. Let $A \subseteq L(S)$, and $S|_A$ be the smallest species tree inferred from S by sequences of the following operations: (1) remove a leaf from S that is labeled by a species from $L(S) \setminus A$, and (2) contract a non-root node of degree 2.

For a gene tree G and a species tree S such that $L(G) \subseteq L(S)$, a *least common ancestor mapping*, or lca-mapping, is a function from the nodes of G to the nodes of S such that $M(g) = s$ if g and s are leaves with the same label, or $M(g) = M(a) + M(b)$ if g is an internal node of G such that a and b are the children of g.

Now, we introduce several cost functions used when reconciling a gene tree G and a species tree S. An internal node $g \in V_G$ is a *gene duplication* if $M(g) = M(g')$ for g' a child of g. The total number of gene duplications is called *duplication cost* and denoted by $D(G, S)$. We define the *deep coalescence* cost function as follows: $DC(G, S) = \sum_{a,b \text{ siblings in } G}(|\pi(M(a), M(b))| - 1)$, where $\pi(x, y)$ is the set of all nodes on the shortest path connecting x and y in S. Note, the standard definition of the *deep coalescence* cost function [2] differs by $1 - |V_G|$ from ours, which is a constant value dependent on G. Next, we define *the loss cost [28]:* $L(G, S) = DC(G, S) + 2 \times D(G, S) - |V_G| + 1$ and *the duplication-loss cost*: $DL(G, S) = D(G, S) + L(G, S)$. For a given gene tree G and a given species tree S all costs mentioned above can be computed in linear time and space [17].

Here, we need a more general construction that will precisely assign partial costs to a node of a species tree. Suppose we are given *a contribution function* $\xi\colon V_G \times V_S \to R$, which, for nodes $g \in V_G$ and $s \in V_S$, assigns a real $\xi(g, s)$ representing a cost contribution of a node g to a node s when reconciling G and S. Then, the cost of reconciling a gene tree G and a species S will be represented in

a general form $\sigma(G,S) = \sum_{g\in G, s\in S} \xi(g,s)$. Let us define the cost contribution functions for our standard cost functions. Let $\mathbf{1}$ be the indicator function, that is, $\mathbf{1}[P] = 1$ if P is satisfied, and $\mathbf{1}[P] = 0$ otherwise. Then,

$$\begin{aligned}
\xi^D(g,s) &= \mathbf{1}[\exists g' \in Ch(g) \text{ and } M(g) = s = M(g')], && (1)\\
\xi^L(g,s) &= \mathbf{1}[\{g',g''\} = Ch(g) \text{ and } s \in \text{In}(M(g'),M(g''))], && (2)\\
\xi^{DC}(g,s) &= \mathbf{1}[\{g',g''\} = Ch(g) \text{ and } M(g) \neq s \in \pi(M(g'),M(g''))], && (3)\\
\xi^{DL}(g,s) &= \xi^D(g,s) + \xi^L(g,s), && (4)
\end{aligned}$$

where $\text{In}(v,w)$ is the set of loss nodes between v and w defined as follows. If $v \leq w$, then $\text{In}(v,w) = \pi(v,w) \setminus \{v\}$. If $w \leq v$, then $\text{In}(v,w) = \pi(v,w) \setminus \{w\}$. Finally, $\text{In}(v,w) = \pi(v,w) \setminus \{v,w,v+w\}$ if v and w are incomparable. In this general setting, one can prove that: $D(G,S) = \sigma^D(G,S)$, $L(G,S) = \sigma^L(G,S)$ and so on [13].

We say that a species tree S is *over a set of species* I if $L(S) = I$. Let Q be a collection of gene trees $G_1, G_2, \ldots, G_n$. By $L(Q)$ we denote the set of all species present in trees of Q. Now, in a natural way we extend the notion of the cost function to collections of gene trees. For a species tree S over $L(Q)$, by $c(Q,S)$ we denote the total cost $\sum_{G\in Q} c(G,S)$.

Problem 2.1. *[OST - Optimal Species Tree] Given a collection of gene trees Q and a cost function c, find a species tree S^{opt} that minimizes the total cost $c(Q,S)$ in the set of all species trees S over $L(Q)$.*

The species tree S^{opt} will be called optimal (for Q and c) and the optimal cost we denote by $c^{opt}(Q)$. A simpler variant of the previous problem is the OOC problem, which is NP-hard under the duplication cost [18].

Problem 2.2. *[OCC - Optimal Cost Computation] Given a collection of gene trees Q and a cost function c, compute the optimal cost $c^{opt}(Q)$.*

3 Methods

We show a dynamic programming solution for the OCC problem, and then provide an extension to the algorithm that solves the OST problem.

Any internal node g of a gene tree G determines a multiset $\{A,B\}$, where A and B are the clusters of children of g. Such a multiset will be called *a rooted split* and denoted by $A|B$. For a collection Q of gene trees, let $r(Q)$ be the multiset of all rooted splits present in trees of G.

Dynamic programming formula. Let Q be a collection of gene trees, c be a cost function, and $Z \subseteq L(Q)$. The dynamic programming formula Δ^c for the OCC problem is defined as follows (the superscript c is omitted in Δ, Γ and Λ):

$$\Delta(Q,Z) = \begin{cases} \Lambda(Q,s) & \text{if } Z = \{s\},\\ \min\limits_{\substack{Z=X\cup Y\\ X\cap Y=\emptyset\\ X,Y\neq\emptyset}} \Delta(Q,X) + \Delta(Q,Y) + \Gamma(Q,X,Y) & \\ & \text{otherwise,}\end{cases} \quad (5)$$

where $\Lambda(Q,s)$ is the total cost contribution of the nodes from Q to a leaf of some species tree over $L(G)$ labeled by s, and $\Gamma(Q,X,Y)$ is the total cost contribution of the nodes from Q to an internal node v of some species tree over $L(G)$ such that the cluster of v is $X \cup Y$, and v has two children whose clusters are X and Y. Observe that under our standard cost functions these definitions do not depend on the choice of the species tree.

The total cost contribution functions for our standard costs are defined as follows:

$$\begin{aligned}
\Lambda(Q,s) &= \sum_{q\in r(Q)} \lambda(q,s)\\
\Gamma(Q,X,Y) &= \sum_{q\in r(Q)} \gamma(q,X,Y),
\end{aligned}$$

where $\lambda^D(A|B,s) = \mathbf{1}[A\cup B = \{s\}]$, $\lambda^L(A|B,s) = 0$, $\lambda^{DL} = \lambda^D$, $\lambda^{DC}(A|B,s) = 0$ and

$$\begin{aligned}
\gamma^D(A_1|A_2,X_1,X_2) &= \mathbf{1}[A_1\cup A_2 \subseteq X_1\cup X_2 \wedge \exists i\colon \neg A_i \subseteq X_1 \wedge \neg A_i \subseteq X_2], && (6)\\
\gamma^L(A_1|A_2,X_1,X_2) &= \mathbf{1}[\exists i,j\colon A_i \subseteq X_j \wedge A_{i+1} \subseteq X_1\cup X_2 \Rightarrow (A_{i+1}\cap X_1 \neq \emptyset \neq A_{i+1}\cap X_2)], && (7)\\
\gamma^{DC}(A_1|A_2,X_1,X_2) &= \mathbf{1}[\exists i,j\colon (\neg A_i \subseteq X_j \Rightarrow A_i \subseteq X_1\cup X_2) \wedge \neg A_{i+1} \subseteq X_1\cup X_2], && (8)\\
\gamma^{DL}(q,X_1,X_2) &= \gamma^D(q,X_1,X_2) + \gamma^L(q,X_1,X_2). && (9)
\end{aligned}$$

For simplicity, in the above equations $i+1$ (and similarly $j+1$) is 1 if $i = 2$. The following lemma establishes relation between a contribution function and the total cost contribution function.

Lemma 3.1. *Let G be a gene tree, c be a cost function, and S be a species tree over $L(G)$. Then, 1) for each leaf v in S labeled by a species s: $\Lambda^c(\{G\},s) = \sum_{g\in V_G} \xi^c(g,v)$. 2) for each internal node v in S whose children have clusters X and Y: $\Gamma^c(\{G\},X,Y) = \sum_{g\in V_G} \xi^c(g,v)$.*

Proof. The proof follows directly for each cost function: D, L, DC and DL and the definitions of λ and γ. □

Theorem 3.2. *Let Q be a collection of gene trees and c be a cost function. Then $\Delta^c(Q,L(Q)) = c^{opt}(Q)$.*

Proof. It is sufficient to prove that for each $Z \subseteq L(Q)$,

$$\Delta^c(Q, Z) = \min_{S \in Tr(Z)} \sum_{\substack{G \in Q, g \in G \\ v \in V_S, c_v \subseteq Z}} \xi^c(g, v),$$

where $Tr(Z)$ is the set of all species trees over $L(Q)$ having a node whose cluster is Z. The proof follows by induction from Lemma 3.1. We omit technical details. □

Solving the OST problem follows directly from the values of Δ; it is sufficient to track which partitions of the cluster Z into X and Y in formula (5) yield the minimal value. Such partitions determine clusters in optimal species trees and they can be used to infer one or all optimal species trees.

Similarly, to solve the OST problem using Δ, our formula can be extended to count the number of all optimal species trees. In addition to tracking which partition of a cluster Z yields the minimal value, we keep track of how many partitions yield such value. Let $\Omega(Q, Z)$ denote the count. Within one partition (X, Y) of Z, $\Omega(Q, X) \times \Omega(Q, Y)$ is the count of all optimal trees under (X, Y), and $\Omega(Q, Z)$ is the sum of $\Omega(Q, X) \times \Omega(Q, Y)$ over all optimal partitions under Z. Due to the recursive nature of Δ, such count equals the number of different species trees whose set of leaves is Z.

Algorithm and its implementation. First, the gene trees present in Q are converted into the multiset of rooted splits $r(Q)$, that is, each split consists of two clusters and its multiplicity. The algorithm that solves the OST problem is based on Formula (5), which can be efficiently implemented by using a bit-vector representation of sets. Each cluster, that is, a set of species from $L(Q)$, is represented as a bit-vector of the size n, where n is the size of $L(Q)$. Each bit-vector operation (see formulas (6)-(9)) requires $O(n/b)$ time, where b is the word size of the platform.

Our algorithm creates the array of 2^n values of Δ. To compute a single $\Delta(Q, Z)$ value, we need to generate all splits of Z. Each split of Z can be computed in constant time by using Gray code [16]. Then, we have $\sum_{2 \le k \le n} \binom{n}{k} 2^k = \Theta(3^n)$ splits of the elements from the powerset of $L(Q)$, therefore the time complexity of the optimal cost computation is $\Theta(3^n mn/b)$, where m is the number of unique rooted splits in $r(Q)$. Clearly, the space complexity is $\Theta(2^n + m)$.

As already mentioned in the previous section, to solve the OST problem, it is sufficient to store exactly one rooted split that yields the minimal score in (5). In such a case the algorithm requires additional $\Theta(2^n)$ units of space (that is, one optimal split per each value in the Δ array). Clearly, the time complexity is the same. To obtain the total number of optimal solutions, we need an additional counter in each element of the Δ array. Updating the counter can be completed in constant time when computing the minimal score in the dynamic programming formula. Hence under the same complexity, we also find the total number of optimal solutions.

In practice, the genes for single gene trees are often sampled from a smaller fractions of the overall set of species from which the genes for all of the gene trees were sampled. A standard solution is to apply *the minus method* [10], where the cost is computed for the species tree contracted to the set of species present in a given gene tree. Formally, for a given cost c, a species tree S, and a gene tree G, such that $L(G) \subseteq L(S)$, instead of computing $c(S, G)$ we compute $c(G, S|_{L(G)})$. If c is the duplication cost, then there is no difference in the dynamic programming algorithm. For the other costs, we need the following changes (proofs are straightforward and omitted for brevity):

- the data structure $r(Q)$ is the multiset of triplets: $\{A|B|L(G)\colon\ G \in Q,\ A|B \text{ is a rooted split from } G\}$; in other words each rooted split is extended with the set of species present in its source gene tree.
- the formula (7) for gene losses is replaced by: $\gamma^L(A_1|A_2|C, X_1, X_2) = \mathbf{1}[\exists i, j\colon A_i \subseteq X_j \wedge (A_{i+1} \subseteq X_1 \cup X_2 \Rightarrow A_{i+1} \cap X_1 \neq \emptyset \neq A_{i+1} \cap X_2) \wedge X_{j+1} \cap C \neq \emptyset]$,
- the formula (8) for deep coalescence is replaced by: $\gamma^{DC}(A_1|A_2|C, X_1, X_2) = \mathbf{1}[\exists i, j\colon (\neg A_i \subseteq X_j \Rightarrow A_i \subseteq X_1 \cup X_2) \wedge \neg A_{i+1} \subseteq X_1 \cup X_2 \wedge X_{j+1} \cap C \neq \emptyset]$.

The time and space complexity remain the same, however, the size of $r(Q)$ (the parameter m) can increase when the minus method is applied. Therefore, the run-time is usually longer for the DC and DL costs than for the D cost as observed in our simulation study. See Fig. 1 for comparison.

4 Experiments

We implemented our algorithms for the different GTP problems as one software program, and evaluated its performance using a simulation study and an empirical study. Our algorithms were implemented in C/C++, and the testing platform was an Intel® Core™ i7 CPU with 2.80GHz.

Simulation Study. We investigated the runtime of our software for simulated instances (gene trees) of GTP problems. Instances of different numbers of taxa were generated using gene duplication and loss, and deep coalescence models. The simulated instances for a fixed number of taxa may vary in their number of rooted splits and therefore affect the runtime of our software.

Instances for the GTP problems based on gene duplication, and gene duplication and loss were generated using a simplified birth-death process described in [1]. First, we randomly generated a species tree in the range of 10 to 22 taxa that was used as a template for evolving gene

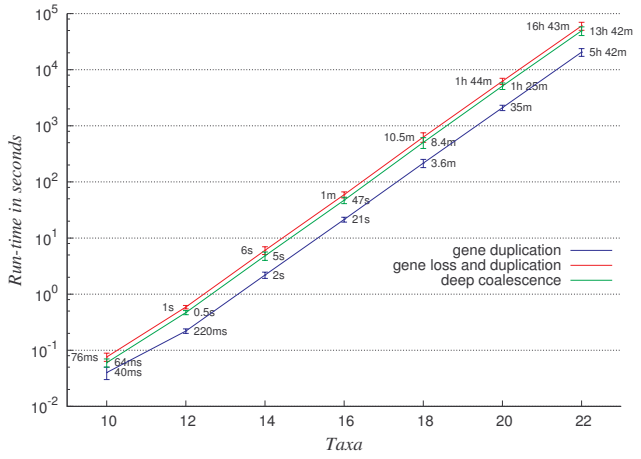

Figure 1: Summary of experiments performed on simulated data sets under standard evolutionary costs. The run-time means and standard deviations are summarized on the logarithmic scale for our standard cost functions: gene duplications (blue curve), gene duplications and losses (red curve) and deep coalescence (green curve). The data sets with less than 16 taxa were resolved in less than 1 minute, while the largest data sets with 22 taxa required up to 17 hours.

trees. For each gene node evolutionary events were simulated based on arbitrarily chosen birth-death parameters. Cloning a sub-tree simulates gene duplication, and removing a sub-tree simulates gene loss.

We performed a total of 70 runs on different simulated data sets for $10, 12, 14, 16, 18, 20$, and 22 taxa (10 runs each). Each instance was generated with 100 gene trees. The runtimes of our software for gene duplication, duplication and loss, and deep coalescence are depicted in Fig. 1. While instances were generated only based on the gene duplication and loss model, runtimes for the deep coalescence cost function were included for comparison purposes. We observed that data sets with up to 16 taxa can be computed in less than 1 minute. Our program found optimal species trees for data sets with up to 22 taxa in less than 17 hours for the GTP problems based on gene duplication, and gene duplication and loss. We further analyzed the number of optimal solutions, and found that each of these generated instances has a unique solution.

We generated instances for the GTP problem based on deep coalescence using Mesquite [20] as outlined in the simulation protocol from Maddison and Knowles [19]. Our effective population size was $10,000$. Similarly to the previous simulation we simulated data sets for 10, 12, 14, 16, 18, 20, and 22 taxa. Our software required about a factor of three more runtime for instances genrated from the deep coalescence model than for instances generated from the duplicaiton loss model. We found that the number of rooted splits differ by a factor of 2.8 to 4.1 between instances that were generated by the two different models of the same taxon size. The deep coalescence instances were also reported to have up to four optimal species trees.

Empirical Study. We re-evaluated the data set from Guigó et al. [14] of 53 gene families from 16 eukaryotes. Page and Charleston [22] identified 12 optimal trees for the GTP problem based on gene duplication and loss events. However, it was an open problem if all optimal species trees were identified [22]. Here, for the first time, we showed with our new software that Page and Charleston have already identified all optimal trees, which took our software less than 42 seconds to compute. Further, for the same data set we computed the number of optimal solutions for the GTP problem based on gene duplications, and found that there are 29 additional optimal trees. This computation took our software only 14 seconds to complete.

Having already computed the exact result for the Guigó *et al.* data set, we used this data set to evaluate the accuracy of the GTP heuristic DupTree [26]. We ran DupTree a thousand times on the Guigó *et al.* data set and found optimal species trees in only 8% of these runs.

5 Conclusion

Our fast bit-vector and Gray encoding algorithms, to the best of our knowledge, provide the largest GTP species trees from non-trivial instances to date. Since the exact algorithms can compute species trees with up to 22 taxa in reasonable time, they can provide new perspectives on phylogenetic questions that previously could only be addressed by heuristics. This is of particular importance, since our empirical study showed that heuristic estimates can be misled in the vast majority of runs on the same data set with as few as 16 taxa. In contrast to heuristics, our algorithms report the number of optimal solutions, which can be beneficial to phylogenetic analyses.

Acknowledgments

We thank Gordon Burleigh for valuable discussions on this work. Support was provided to PG by the grant of MNiSW #N N301 065236, to OE by NSF (#0830012 and #106029), and PG and OE by NCN #2011/01/B/ST6/02777.

References

[1] L. Arvestad, A. Berglund, J. Lagergren, and B. Sennblad. "Bayesian gene/species tree reconciliation and orthology analysis using MCMC". *Bioinformatics*, 19(suppl 1):i7–i15, 2003.

[2] M. S. Bansal, J. G. Burleigh, and O. Eulenstein. "Efficient genome-scale phylogenetic analysis under the

duplication-loss and deep coalescence cost models". *BMC Bioinf.*, 11(Suppl 1):S42, 2010.

[3] M. S. Bansal and R. Shamir. "A note on the fixed parameter tractability of the gene-duplication problem". *IEEE/ACM Trans. Comput. Biol. Bioinf.*, 8(3):848–850, 2011.

[4] G. Blin, P. Bonizzoni, R. Dondi, R. Rizzi, and F. Sikora. "Complexity insights of the minimum duplication problem". *SOFSEM 2012: Theory and Practice of Computer Science*, pages 153–164, 2012.

[5] D. Brown and I. Harrower. "Integer programming approaches to haplotype inference by pure parsimony". *IEEE/ACM Trans. Comput. Biol. Bioinf.*, 3(2):141–154, 2006.

[6] W.-C. Chang, J. G. Burleigh, D. Fernández-Baca, and O. Eulenstein. "An ILP solution for the gene duplication problem". *BMC Bioinf.*, 12(Suppl 1):S14, 2011.

[7] R. Chaudhary, M. S. Bansal, A. Wehe, D. Fernández-Baca, and O. Eulenstein. "iGTP: A software package for large-scale gene tree parsimony analysis". *BMC Bioinf.*, 11(1):574, 2010.

[8] M. Chimani, S. Rahmann, and S. Böcker. "Exact ILP solutions for phylogenetic minimum flip problems". In *Proceedings of ACM-BCB 2010*, pages 147–153. ACM, 2010.

[9] J. Cotton and R. Page. "Going nuclear: gene family evolution and vertebrate phylogeny reconciled". *Proc. R. Soc. London, Ser. B*, 269(1500):1555–1561, 2002.

[10] J. Cotton and M. Wilkinson. "Majority-rule supertrees". *Syst. Biol.*, 56(3):445–452, 2007.

[11] J. Doyon and C. Chauve. "Branch-and-bound approach for parsimonious inference of a species tree from a set of gene family trees". *Software Tools and Algorithms for Biological Systems*, pages 287–295, 2011.

[12] M. Goodman, J. Czelusniak, G. Moore, A. Romero-Herrera, and G. Matsuda. "Fitting the gene lineage into its species lineage, a parsimony strategy illustrated by cladograms constructed from globin sequences". *Syst. Biol.*, 28(2):132–163, 1979.

[13] P. Górecki and J. Tiuryn. "Inferring phylogeny from whole genomes". *Bioinformatics*, 23(2):e116–e122, 2007.

[14] R. Guigo, I. Muchnik, and T. Smith. "Reconstruction of ancient molecular phylogeny". *Mol. Phylogenet. Evol.*, 6(2):189–213, 1996.

[15] D. Gusfield, Y. Frid, and D. Brown. "Integer programming formulations and computations solving phylogenetic and population genetic problems with missing or genotypic data". *Computing and Combinatorics*, pages 51–64, 2007.

[16] D. Knuth. *The Art of Computer Programming, Volume 4, Fascicle 2: Generating All Tuples and Permutations*. Addison-Wesley Professional, 2005.

[17] B. Ma, M. Li, and L. Zhang. "On reconstructing species trees from gene trees in term of duplications and losses". In *Proceedings of RECOMB 98*, pages 182–191. ACM, 1998.

[18] B. Ma, M. Li, and L. Zhang. "From gene trees to species trees". *SIAM Journal on Computing*, 30(3):729–752, 2001.

[19] W. Maddison and L. Knowles. "Inferring phylogeny despite incomplete lineage sorting". *Syst. Biol.*, 55(1):21–30, 2006.

[20] W. Maddison and D. Maddison. "Mesquite: a modular system for evolutionary analysis". *Evolution*, 62(5):1103–1118, 2008.

[21] R. Page. "Extracting species trees from complex gene trees: reconciled trees and vertebrate phylogeny". *Mol. Phylogenet. Evol.*, 14(1):89–106, 2000.

[22] R. Page and M. Charleston. "Reconciled trees and incongruent gene and species trees". *Mathematical Hierarchies in Biology*, 37:57–70, 1997.

[23] M. Sanderson and M. McMahon. "Inferring angiosperm phylogeny from EST data with widespread gene duplication". *BMC Evol. Biol.*, 7(Suppl 1):S3, 2007.

[24] J. Slowinski and R. Page. "How should species phylogenies be inferred from sequence data?". *Syst. Biol.*, 48(4):814–825, 1999.

[25] C. Than and L. Nakhleh. "Species tree inference by minimizing deep coalescences". *PLoS Comput. Biol.*, 5(9):e1000501, 2009.

[26] A. Wehe, M. S. Bansal, J. G. Burleigh, and O. Eulenstein. "DupTree: a program for large-scale phylogenetic analyses using gene tree parsimony". *Bioinformatics*, 24(13):1540–1541, 2008.

[27] A. Wehe, J. G. Burleigh, and O. Eulenstein. "Algorithms for knowledge-enhanced supertrees". In *Proceedings of ISBRA 2012*, pages 263–274, 2012.

[28] L. Zhang. "From gene trees to species trees II: Species tree inference by minimizing deep coalescence events". *IEEE/ACM Trans. Comput. Biol. Bioinf.*, 8(6):1685–1691, 2011.