A probabilistic algorithm for gene-species reconciliation with segmental duplications

Yao-Ban Chan, Celine Scornavacca, Michael Charleston

A probabilistic algorithm for gene-species reconciliation with segmental duplications

Yao-ban Chan[1,2,*], Celine Scornavacca[3], and Michael Charleston[4]

[1] Melbourne Integrative Genomics

The University of Melbourne, Melbourne, Victoria, Australia

[2] School of Mathematics and Statistics

The University of Melbourne, Melbourne, Victoria, Australia

[3] Institut des Sciences de l'Evolution de Montpellier

Université de Montpellier, CNRS, IRD, EPH, Montpellier, France

[4] School of Natural Sciences

University of Tasmania, Australia

*To whom correspondence should be addressed;

E-mail: yaoban@unimelb.edu.au

September 22, 2025

Keywords: reconciliation, segmental duplication, Boltzmann distribution

Abstract:

Reconciliations are a mathematical tool to compare the phylogenetic trees of genes to the species that contain them, accounting for events such as gene duplication and loss. Traditional reconciliation methods have predominantly relied on parsimony to infer gene-only evolutionary events, and usually make the hypothesis that genes evolve independently. Recently, more advanced models have been developed that account for complex gene interactions stemming from phenomena such as segmental duplications, where multiple genes undergo simultaneous duplication. In this paper, we study the NP-hard problem of reconciling gene trees to a species tree with segmental duplications, without the aid of synteny information. We address this problem by proposing a novel probabilistic approach, imposing a Boltzmann distribution over the space of reconciliations. This allows for a Gibbs sampling-like Markov chain Monte Carlo algorithm that uses simulated annealing to effectively find or approximate the most parsimonious

reconciliation, as demonstrated through rigorous simulations and re-analysis of empirical datasets. Our findings present a promising new framework for addressing NP-hard reconciliation challenges in phylogenetics, enhancing our understanding of gene evolution and its relationship to species evolution.

1 Introduction

One of the goals of evolutionary biology is to understand the processes through which species change over time through mechanisms such as natural selection, genetic drift, mutation, and gene flow. Since species harbour a large number of genes, each of which have their own histories, comparing the phylogenetic trees of genes alongside those of the species that contain them can provide insights into the evolution of both genes and species.

One way of comparing gene and species trees is through reconciliations, which map the gene tree into the species tree by explicitly inferring a number of gene-only evolutionary events. There are two main paradigms for the reconstruction of reconciliations. Parsimony considers events to be relatively unlikely, and so attempts to find a reconciliation which minimises a cost function, which is a weighted sum of the number of events, with each event type assigned a (potentially) different weight. On the other hand, probabilistic methods attempt to find a reconciliation with a maximised likelihood under a statistical model of evolution. Because of the complexity and the time-consuming nature of calculating these likelihoods, parsimony methods are still largely prevalent in this area.

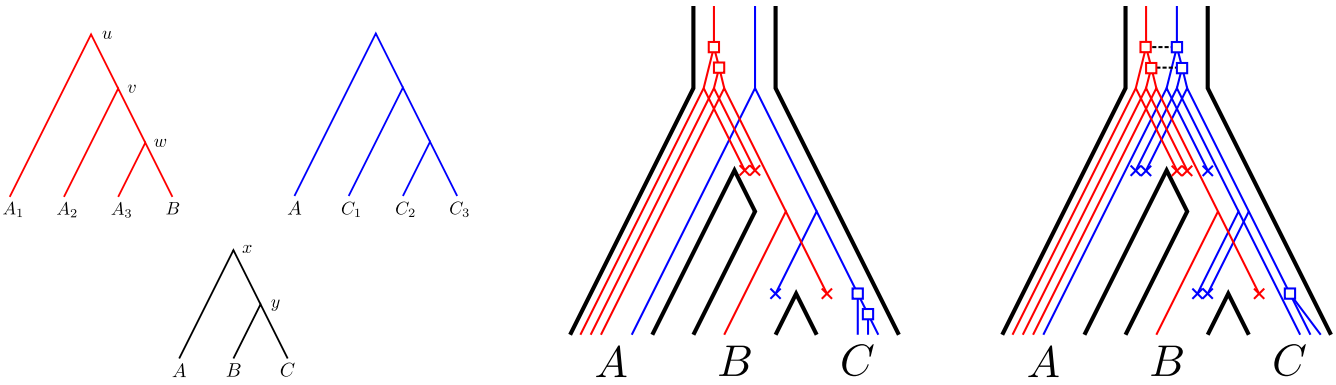
The first reconciliation model (Goodman et al., 1979) included only (individual) gene duplication and gene loss events (the DL model). For this model, it was shown that the most parsimonious reconciliation (MPR) can easily be found with the so-called *LCA mapping*, where each ancestral gene is mapped to the lowest common ancestor of the species that its descendants appear in. The LCA mapping minimises both the number of duplications and the number of losses, and thus any combination of these two (Górecki and Tiuryn, 2006), and can be calculated in linear time (Zhang, 1997).

The first extension to the DL model included horizontal gene transfers as well as duplications and losses (the DTL model), and for a dated species tree finding an MPR is polynomial-time (Doyon et al., 2011), but not linear. Over time, reconciliation models (and corresponding methods to find an MPR) have become more sophisticated, including an ever-growing number of gene evolutionary events. These include gene conversions (Hasić and Tannier, 2019a), transfers with replacement (Hasić and Tannier, 2019b), "failure to diverge" (Drinkwater et al., 2016), and paralog exchange (Chan and Robin, 2019). An increasing number of models have also been formulated to include incomplete lineage sorting (Rasmussen and Kellis, 2012; Wu et al., 2014; Chan et al., 2017; Li et al., 2021), which also produces gene-species discordance despite not being an evolutionary 'event'. A recent review of developments in the field can be found in Menet et al. (2022).

Most reconciliation models consider the reconciliation of a single gene tree to a species tree, necessarily consid-

ering multiple gene families independent of each other. However, the evolution of genes are not independent, as genes located near to each other on the chromosome have related histories. This is because of events that may affect not just one gene, but a whole segment of chromosome, potentially involving many genes at once. One such event is segmental duplication, where a chromosome segment duplicates, resulting in the simultaneous duplication of all genes in the segment. This can occur up to the scale of whole chromosomes or even genomes, and whole genome duplication is an important driver in polyploidy. Other such events include segmental transfers or losses.

An early attempt to detect correlated evolution used statistical tests on single-tree reconciliations (Chan et al., 2013). More recently, Dondi et al. (2019) formulated the problem of reconciliation with segmental duplications, where a number of gene trees are simultaneously reconciled to a species tree, and duplications appearing on the same species branch are considered (if possible) to be a single segmental duplication (see Figure 1 for an example). They showed that this problem is NP-hard for its unrestricted version (constrained versions can be solved in polynomial time, see for example Paszek and Górecki, 2018), and gave an algorithm to solve it that is exponential in the size of the problem, but fixed-parameter tractable. Alternative scoring schemes, such as the maximal path or MP score, also exist and are solvable in polynomial time (Paszek et al., 2020).



(a) Two gene trees (red, blue) and a species tree (black)

(b) Reconciliation without segmental duplication

(c) Reconciliation with segmental duplication

Figure 1: An example of reconciliation with segmental duplications. Reconciling the gene trees individually infers 4 duplications (squares); however, reconciling the gene trees simultaneously infers only 3 segmental duplications but a larger number of losses (crosses), which can be optimal if duplications are sufficiently costly with respect to loss events.

NP-hard problems are a recurring feature in reconciliations (Menet et al., 2022), and indeed much of phylogenetics. They often occur when gene lineages are dependent in some way, such as in the segmental duplications problem, as this precludes the dynamic programming paradigm that is traditionally employed to solve the MPR problem (for example, for the DTL model). It is thus of interest to find a new way to approach these types of problems that does not employ dynamic programming.

In the model of Dondi et al. (2019), a duplication in any gene family can form a segmental duplication with any other gene family. Arguably, a more realistic but more complicated model explicitly takes into account the relative positions of genes on the chromosome, if these are known and conserved enough. The genes can then be

considered to form *synteny blocks* which constrain the possibilities for segmental duplications; furthermore, the blocks themselves can join or separate, corresponding to the movement of genetic loci in the chromosome (Delabre et al., 2020; Anselmetti et al., 2022).

In this paper, we consider the case where no synteny information is provided, and we focus on the original problem of reconciliation with segmental duplications from Dondi et al. (2019). We develop a new approach to this problem by imposing a Boltzmann distribution on the set of possible reconciliations; that is, each reconciliation occurs with probability proportional to the exponential of its negative cost, weighted by a *temperature* parameter. The Boltzmann distribution on reconciliations has been used before (Chauve et al., 2015) as a tool to access and study different regions of the reconciliation space. However, previous work used dynamic programming (DP) to sample from the Boltzmann distribution; if this is feasible, then it is also possible to use dynamic programming to find an MPR directly, but no polynomial-time DP algorithm exists for the segmental duplications problem.

Here, we use the Boltzmann distribution to find the most parsimonious reconciliation directly. By use of a Gibbs sampling-like Markov chain Monte Carlo approach combined with simulated annealing, we develop a new algorithm to find an MPR for the segmental duplication model. This is a unique approach in that it is a probabilistic algorithm to solve a discrete optimisation problem; while it is technically a heuristic, in practice it works very well to find an exact solution. We demonstrate the effectiveness of this algorithm on simulated data, showing that it outperforms the existing algorithm of Dondi et al. (2019), in that it can find a better reconciliation on average, and runs faster for larger problem instances. We also use our algorithm to re-analyse datasets from Guigó et al. (1996), recovering a previously found optimal reconciliation while enabling a more thorough exploration of the solution space, and Butler et al. (2009), showing the potential of our algorithm on large datasets. This work provides a new paradigm to address NP-hard reconciliation problems.

2 Preliminaries

Given a tree T, the sets of its nodes, edges, and leaves are denoted $V(T), E(T)$, and $L(T)$ respectively. For a collection of trees, or a gene forest, denoted as $\mathcal{G}$, we define the sets $V(\mathcal{G})$, $E(\mathcal{G})$, and $L(\mathcal{G})$ as the unions of the respective sets $V(\cdot)$, $E(\cdot)$, and $L(\cdot)$ taken across all gene trees in $\mathcal{G}$.

For a node $u \in V(T)$, we denote its unique parent by u_p, and its two children by $\{u_l, u_r\}$. We define T_u to be the subtree of T rooted at u.

For two nodes $u, v \in V(T)$, we write $u \leq v$ (resp. $u < v$) if v lies on the unique path from the root of T to u (resp., and $u \neq v$). (Note that the direction of this inequality is not universally consistent in the literature.) We say in this case that u is a descendant of v, and v is an ancestor of u, and define $d(u, v)$ to be the number of edges on the unique path from u to v. If $u \leq v$ or $v \leq u$, we say that u and v are comparable; otherwise, they are incomparable.

For a set of leaf nodes $C \subseteq L(T)$, the lowest common ancestor of C, denoted $LCA(C)$, is the unique minimal

node (i.e., no other node is descendant to it) such that $LCA(C) \geq v$ for all $v \in C$.

A gene (resp. species) tree is a binary tree whose leaves represent extant genes (resp. species). Conventionally and for clarity, we will refer to gene nodes and species vertices (even though they are the same concept in graph theory). Given a gene tree G and species tree S, each extant gene is associated with the species that contains it by the species labelling $s : L(G) \to L(S)$. For a gene node $u \in V(G)$, we define the LCA mapping $sLCA(u) = LCA(s(L(G_u)))$; that is, the species vertex that is the LCA of the species leaves that contain all genes descending from u.

We now reproduce a few definitions from Dondi et al. (2019). Firstly, let $\mathbb{S}, \mathbb{D}, \mathbb{C}$ be symbols representing the three possible events at a gene node: speciation, duplication, and contemporary event respectively.

Definition 1 (Reconciliation, equivalent to that in Dondi et al., 2019)**.** *Given a gene forest $\mathcal{G}$ and a corresponding species tree S, a* reconciliation *between $\mathcal{G}$ and S is a function $\alpha : V(\mathcal{G}) \to V(S) \times \{\mathbb{S}, \mathbb{D}, \mathbb{C}\}$ that maps each node $u \in V(\mathcal{G})$ to a tuple $\alpha(u) = (\alpha^{\mathrm{v}}(u), \alpha^{\mathrm{e}}(u))$, such that:*

- *if u is a leaf node of $\mathcal{G}$, then $\alpha^{\mathrm{e}}(u) = \mathbb{C}$ and $\alpha^{\mathrm{v}}(u) = s(u)$;*
- *if u is not a leaf node of $\mathcal{G}$, then either:*
  - *$\alpha^{\mathrm{e}}(u) = \mathbb{S}$ and $\alpha^{\mathrm{v}}(u_l) \leq \alpha^{\mathrm{v}}(u)_l$ and $\alpha^{\mathrm{v}}(u_r) \leq \alpha^{\mathrm{v}}(u)_r$, or vice versa; or*
  - *$\alpha^{\mathrm{e}}(u) = \mathbb{D}$ and $\alpha^{\mathrm{v}}(u_l), \alpha^{\mathrm{v}}(u_r) \leq \alpha^{\mathrm{v}}(u)$.*

Here, each leaf node in the gene tree is mapped to the species that contains it with a contemporary event, while each internal gene tree node u is mapped to the species vertex $\alpha^{\mathrm{v}}(u)$ (if a speciation) or the branch above it (if a duplication); $\alpha^{\mathrm{e}}(u)$ determines the event type associated with u.

For each species vertex $x \in V(S)$, let $\mathcal{G}(\alpha, x)$ be the gene node set $\{v \in V(\mathcal{G}) : \alpha(v) = (x, \mathbb{D})\}$. Let $n_{SD}(\alpha, x)$ be the minimum size of a partition of nodes of $\mathcal{G}(\alpha, x)$, such that no two nodes in the same subset are comparable in $\mathcal{G}$. Equivalently, $n_{SD}(\alpha, x)$ is the height of $\mathcal{G}(\alpha, x)$. Then the number of segmental duplications in α is given by $n_{SD}(\alpha) = \sum_{x \in V(S)} n_{SD}(\alpha, x)$.

For each gene tree edge $(u, v) \in E(\mathcal{G})$, let $l(\alpha, u, v) = d(\alpha^{\mathrm{v}}(u), \alpha^{\mathrm{v}}(v))$ if $\alpha^{\mathrm{e}}(u) = \mathbb{D}$, and $d(\alpha^{\mathrm{v}}(u), \alpha^{\mathrm{v}}(v)) - 1$ otherwise. This can be considered as the number of losses induced on the edge (u, v) by α; one loss must be inferred for each species vertex that the gene tree edge passes though. Then the number of losses in α is given by $l(\alpha) = \sum_{(u,v) \in E(\mathcal{G})} l(\alpha, u, v)$.

We can now define the most parsimonious reconciliation problem.

Problem 1 (Most parsimonious reconciliation (MPR), equivalent to that in Dondi et al., 2019)**.**

***Instance**: A gene forest $\mathcal{G}$, a species tree S, and costs $\delta, \lambda \geq 0$ for duplications and losses respectively.*

***Output**: A reconciliation α between $\mathcal{G}$ and S that minimises $c_{SD}(\alpha) := \delta \cdot n_{SD}(\alpha) + \lambda \cdot l(\alpha)$.*

To illustrate these definitions, consider Figure 1 again. Here, $\mathcal{G}$ contains the two gene trees in Figure 1a (red and blue), and S is the species tree (black). Let α be the reconciliation shown in Figure 1c. Then (for example):

- $\alpha(A_1) = (A, \mathbb{C})$;
- $\alpha(u) = \alpha(v) = (x, \mathbb{D})$;
- $\alpha(w) = (x, \mathbb{S})$.

The number of segmental duplications at the species root vertex x is $n_{SD}(\alpha, x) = 2$, and the total number is $n_{SD}(\alpha) = 3$. The number of losses on the gene branch (w, B) is $l(\alpha, w, B) = 1$, and the total number (across both gene trees) is $l(\alpha) = 8$. This gives a total cost of $c_{SD}(\alpha) = 3\delta + 8\lambda$.

3 A probabilistic algorithm for the MPR problem

In this section we introduce a probabilistic algorithm to solve Problem 1. We use a Gibbs sampling-like Markov chain Monte Carlo approach, combined with simulated annealing. We start by introducing a basic version of our method (Section 3.1), which we will optimise in Sections 3.2 and 3.3.

3.1 MCMC sampling from the Boltzmann distribution

Given a gene forest $\mathcal{G}$ and a species tree S, we impose a Boltzmann distribution on the space of reconciliations between $\mathcal{G}$ and S, so that a reconciliation with a cost of c is assigned the probability

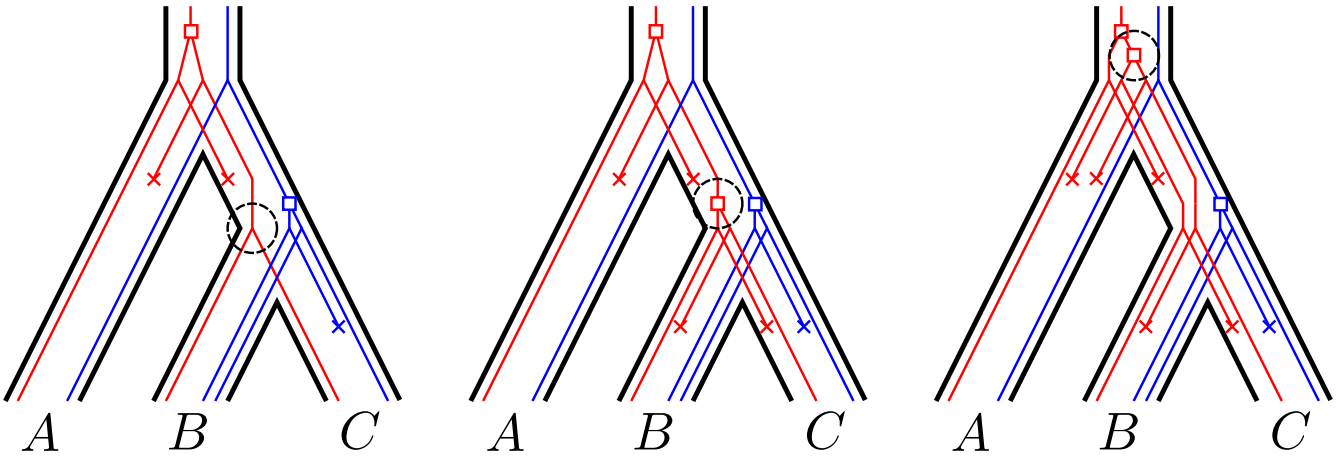
$$Pr(\alpha) \propto e^{-c_{SD}(\alpha)/T}, \tag{1}$$

where T is a *temperature* parameter[1]. When the temperature T is high, all reconciliations have similar probabilities, and the distribution approaches the uniform distribution as $T \to \infty$. Conversely, when the temperature T is low, the most parsimonious reconciliation(s) have a much higher probability than others, and the distribution converges to the uniform distribution on the MPRs only as $T \to 0$.

We sample from this distribution with a Markov chain Monte Carlo walk using Gibbs sampling. Pseudocode for the algorithm is given in Algorithms 1 and 2, which can be understood as follows.

We start with an initial reconciliation, which we take here to be the LCA mapping from $\mathcal{G}$ to S. We then consider one node u of the gene forest $\mathcal{G}$, chosen uniformly at random. We resample the image of u, i.e., $\alpha(u)$. The

[1] As T has no physical meaning here, it absorbs Boltzmann's constant k_B in the traditional formulation of the Boltzmann distribution.



(a) Original reconciliation (b) Cost increases by $\delta + 2\lambda$ (c) Cost increases by $\delta + 3\lambda$

Figure 2: An example of Gibbs sampling. The circled gene tree node is chosen to be resampled; the probabilities of the three configurations shown here are proportional to $1, e^{-(\delta+2\lambda)/T}$, and $e^{-(\delta+3\lambda)/T}$ respectively. There are no other possibilities as the node must be resampled in a valid position (at or above its LCA and below its parent).

set of possible new images for u is constrained by the images of its parent and children, and formally defined as

$$M_u = \{(x \in V(S), \mathbb{D}) : x \geq sLCA(u), \alpha^{\mathrm{v}}(u_l), \alpha^{\mathrm{v}}(u_r) \text{ and } (x \leq \alpha^{\mathrm{v}}(u_p) \text{ if } \alpha^{\mathrm{e}}(u_p) = \mathbb{D} \text{ or } x < \alpha^{\mathrm{v}}(u_p) \text{ if } \alpha^{\mathrm{e}}(u_p) = \mathbb{S})\}$$
$$\cup \{(sLCA(u), \mathbb{S}) \text{ if } sLCA(u) > \alpha^{\mathrm{v}}(u_l), \alpha^{\mathrm{v}}(u_r)\}.$$

(Note that the informal description of M_u in Algorithm 1 considers a duplication on a species branch to be "above" a speciation on the same branch, as well as a duplication corresponding to a child gene node on the same branch.)

We resample from this set in proportion to the probability of the resulting reconciliation, conditioned on the existing images of the other nodes in $\mathcal{G}$. Because of the nature of the Boltzmann distribution, these conditional probabilities only depend on the difference in costs between the original and the proposed reconciliations. This can be computed quickly by calculating the differences in the numbers of segmental duplications and losses between the two reconciliations, which only depend on u and its new and original neighbours in the gene and species trees. An example of this procedure is given in Figure 2.

We then repeat this re-sampling procedure over many iterations. For a fixed temperature, this produces a sequence of reconciliations whose distribution converges to the target Boltzmann distribution. To find the most parsimonious reconciliation, we decrease the temperature linearly to near 0 from a fixed initial value (simulated annealing). This enables us to sample only from the MPRs.

Algorithm 1: Given a gene forest $\mathcal{G}$, species tree S, costs δ, λ, and a number of iterations $t_{\max}$, return an optimal reconciliation between $\mathcal{G}$ and S.

1: Set α to be the LCA mapping from $\mathcal{G}$ to S
2: **for** $(t = 1, \ldots, t_{\max})$ **do**
3: $T \leftarrow T_0 \left(1 - \frac{t-1}{t_{max}}\right)$ ▷ T decreases linearly from T_0 to $\frac{T_0}{t_{max}}$
4: choose an internal node u of $\mathcal{G}$ at random

5: set M_u to be the set of all possible images of u: duplications at or above its LCA mapping and above the images of its children, and below the image of its parent...
6: ...and a speciation at its LCA mapping if that is above the images of its children
7: **for** (each image m in M_u) **do**
8: set α_m to be α with the image of u set to m
9: calculate $c_{SD}(\alpha_m)$ with Algorithm 2
10: resample $\alpha(u)$ from $m \in M_u$ with probability proportional to $e^{-c_{SD}(\alpha_m)/T}$
11: **return**(α)

Algorithm 2: Given a gene forest $\mathcal{G}$, species tree S, costs δ, λ, and two reconciliations α and α_m, return $c_{SD}(\alpha_m)$. Assumes that $c_{SD}(\alpha)$ is known, and α and α_m only differ in the image of one node u such that $\alpha_m(u) = m$.

1: $c_{SD}(\alpha_m) \leftarrow c_{SD}(\alpha)$
2: **if** $(m = \alpha(u))$ **then** ▷ No change to α
3: **return**$(c_{SD}(\alpha_m))$
4: **if** $(\alpha_m^{\mathrm{v}}(u) \geq \alpha^{\mathrm{v}}(u))$ **then** ▷ u is raised (placed closer to the root) in α_m
5: add losses to $c_{SD}(\alpha_m)$ equal to the distance between the two images $d(\alpha_m^v(u), \alpha^{\mathrm{v}}(u))$, increased by 2 if u was originally a speciation in α
6: **if** $(\alpha_m^{\mathrm{v}}(u) \leq \alpha^{\mathrm{v}}(u))$ **then** ▷ u is lowered in α_m
7: subtract losses from $c_{SD}(\alpha_m)$ equal to the distance between the two images $d(\alpha_m^v(u), \alpha^{\mathrm{v}}(u))$, increased by 2 if u is now a speciation in α_m
8: **if** $(n_{SD}(\alpha_m, \alpha_m^{\mathrm{v}}(u)) > n_{SD}(\alpha, \alpha_m^{\mathrm{v}}(u)))$ **then** ▷ more duplications in α_m at $\alpha_m^{\mathrm{v}}(u)$
9: add one duplication to $c_{SD}(\alpha_m)$
10: **if** $(n_{SD}(\alpha_m, \alpha^{\mathrm{v}}(u)) < n_{SD}(\alpha, \alpha^{\mathrm{v}}(u)))$ **then** ▷ fewer duplications in α_m at $\alpha^{\mathrm{v}}(u)$
11: subtract one duplication from $c_{SD}(\alpha_m)$
12: **return**$(c_{SD}(\alpha_m))$

The basic algorithm given in Algorithm 1 is a general-purpose approach that can be broadly applied to many different models. By itself, it is sufficient to solve the MPR problem for segmental duplications for small instances, but for larger instances it can take a large number of iterations to converge. We thus include two improvements specifically tailored to the segmental duplications problem.

3.2 Forced speciations

It is possible to determine some nodes that must be speciations in any MPR. We call such nodes *forced speciations*. The conditions are given in the following definition.

Definition 2 (Forced speciation)**.** *Let $u \in V(\mathcal{G})$ be an internal node of $\mathcal{G}$. If $sLCA(u_l), sLCA(u_r) < sLCA(u)$, and all children of u are either forced speciations or leaf nodes of $\mathcal{G}$, then u is a forced speciation.*

For example, the circled node in Figure 2a above is a forced speciation. Although this definition is recursive, forced speciations can be easily found by considering gene nodes from the bottom up. The following Lemma shows that (as implied by the name) all forced speciations must be assigned as speciations in any MPR.

Lemma 1. *Let α be an MPR between $\mathcal{G}$ and S. If u is a forced speciation, then $\alpha(u) = (sLCA(u), \mathbb{S})$.*

Proof. Suppose there exists some MPR α between $\mathcal{G}$ and S where the result of the Lemma does not hold. Let u be a minimal forced speciation such that $\alpha(u) \neq (sLCA(u), \mathbb{S})$.

Consider the reconciliation α', where $\alpha'(u) = (sLCA(u), \mathbb{S})$ and $\alpha'(v) = \alpha(v)$ for all $v \neq u$. Since u_l and u_r are either leaf nodes or forced speciations that (by construction) are mapped to their LCAs as speciations, this is a valid reconciliation. Now α' has at most the same number of segmental duplications as α, $d(\alpha^{\mathrm{v}}(u), sLCA(u)) + 1$ fewer losses on each of the branches (u, u_l) and (u, u_r), and $d(\alpha^{\mathrm{v}}(u), sLCA(u))$ more losses on the branch (u_p, u) (if it exists). Thus $c_{SD}(\alpha') < c_{SD}(\alpha)$, a contradiction. □

To incorporate this result into our algorithm, we first note that all forced speciations are assigned as speciations in our initial mapping (the LCA mapping). Then, at line 4 of Algorithm 1, we only choose u from nodes that are not forced speciations. We note that this modifies the resulting Boltzmann distribution, so that it no longer covers all possible reconciliations; however, this is inconsequential when aiming at finding an MPR.

3.3 Larger moves: multiple node move

In addition to the basic move of a single gene node (the "single node move"), we also include a "multiple node move" that can move multiple gene nodes at once, but only by the distance of a single species vertex. This helps the convergence of the MCMC algorithm by allowing multiple nodes that are involved in the same segmental duplication to be moved together; otherwise they have to be moved one at a time, initially increasing the number of segmental duplications.

For this move, we first choose to either move nodes up (closer to the root) or down (closer to the tips) with probability $\frac{1}{2}$. There are two cases:

- **Up move**. Choose a vertex uniformly at random from all non-root vertices of S where there are at least 2 duplications assigned by α; call this vertex x. Let A be the set of all gene nodes $u \in V(\mathcal{G})$ with $\alpha(u) = (x, \mathbb{D})$ such that u is either a root, or $\alpha^{\mathrm{v}}(u_p) \geq x_p$ and $\alpha(u_p) \neq (x_p, \mathbb{S})$. Intuitively, A can be thought of as the set of all nodes mapped to x that can be "moved upwards" because they are not constrained by their parents. If $|A| < 2$, we automatically reject the move.

  If not, we choose an integer κ from the interval $[2, |A|]$ with some probability mass function (pmf) $g_{|A|}$, and then choose κ nodes uniformly at random from A. We move the image of all chosen nodes to x_p as duplications. We accept this move using the Metropolis-Hastings (M-H) probability, which we calculate below (we first describe the corresponding down move).

  In this paper, we use $g_n(\kappa) \propto \kappa$ for $\kappa \in [2, n]$. This (gently) encourages more nodes to be moved by placing larger weights on larger sets.

- **Down move**. Choose a vertex uniformly at random from all non-leaf vertices of S where there are at least 2 duplications assigned by α; call this vertex x. Choose a child of x as a destination with probability $\frac{1}{2}$ (suppose

that we choose x_l without loss of generality). Let B be the set of all gene nodes $u \in V(\mathcal{G})$ with $\alpha(u) = (x, \mathbb{D})$ such that $\alpha^{\mathrm{v}}(u_l), \alpha^{\mathrm{v}}(u_r) \leq x_l$. Intuitively, B can be thought of as the set of all nodes mapped to x that can be "moved downwards" to x_l because they are not constrained by their children. If $|B| < 2$, we automatically reject the move.

If not, we choose an integer κ from the interval $[2, |B|]$ with pmf $g_{|B|}$ (where g is as above), and then choose κ nodes uniformly at random from B. We move the image of all chosen nodes to x_l as duplications. We accept this move using the M-H probability, calculated below.

We note that we automatically reject moves that could remap only a single gene node so as to not have different move types be reversible by each other, e.g., a multiple node move can only be reversed by another multiple node move.

The corresponding M-H probabilities are calculated as follows. Let $f(\alpha)$ be the target Boltzmann distribution, as defined in Equation 1. For convenience, let $\chi(\alpha)$ be the number of non-root species vertices x such that at least two gene nodes are mapped by α to $(x, \mathbb{D})$, i.e., the number of species vertices that can be chosen for an up move. Additionally, let $\gamma(\alpha)$ be the number of non-*leaf* species nodes equivalently mapped by α, i.e., the number of species vertices that can be chosen for a down move.

- **Up move**. Denote the resulting reconciliation by α'. Let B' be the set B for the corresponding down move applied to α' at x_p. Then the acceptance probability is the minimum of 1 and:

$$\begin{aligned}\frac{f(\alpha')Pr(\alpha' \to \alpha)}{f(\alpha)Pr(\alpha \to \alpha')} &= \frac{\frac{1}{Z}e^{-c_{SD}(\alpha')/T}\frac{1}{\gamma(\alpha')}\frac{1}{2}g_{|B'|}(\kappa)\frac{1}{\binom{|B'|}{\kappa}}}{\frac{1}{Z}e^{-c_{SD}(\alpha)/T}\frac{1}{\chi(\alpha)}g_{|A|}(\kappa)\frac{1}{\binom{|A|}{\kappa}}} \\ &= \frac{1}{2}\frac{\chi(\alpha)}{\gamma(\alpha')}\frac{g_{|B'|}(\kappa)}{g_{|A|}(\kappa)}\frac{\binom{|A|}{\kappa}}{\binom{|B'|}{\kappa}}e^{-(c_{SD}(\alpha')-c_{SD}(\alpha))/T},\end{aligned}$$

  where Z is the normalising factor for the Boltzmann distribution. (Note that there is a factor of $\frac{1}{2}$ for the down move, corresponding to the choice of the child of x, that is not present for the corresponding up move.)

- **Down move**. Denote the resulting reconciliation by α'. Let A' be the set A for the corresponding up move applied to α' at x_l. Then the acceptance probability is the minimum of 1 and:

$$\begin{aligned}\frac{f(\alpha')Pr(\alpha' \to \alpha)}{f(\alpha)Pr(\alpha \to \alpha')} &= \frac{\frac{1}{Z}e^{-c_{SD}(\alpha')/T}\frac{1}{\chi(\alpha')}g_{|A'|}(\kappa)\frac{1}{\binom{|A'|}{\kappa}}}{\frac{1}{Z}e^{-c_{SD}(\alpha)/T}\frac{1}{\gamma(\alpha)}\frac{1}{2}g_{|B|}(\kappa)\frac{1}{\binom{|B|}{\kappa}}} \\ &= 2\frac{\gamma(\alpha)}{\chi(\alpha')}\frac{g_{|A'|}(\kappa)}{g_{|B|}(\kappa)}\frac{\binom{|B|}{\kappa}}{\binom{|A'|}{\kappa}}e^{-(c_{SD}(\alpha')-c_{SD}(\alpha))/T}.\end{aligned}$$

For the g function that we use, we have

$$\frac{g_{n'}(\kappa)}{g_n(\kappa)} = \frac{\frac{\kappa}{n'(n'+1)/2-1}}{\frac{\kappa}{n(n+1)/2-1}} = \frac{n^2+n-2}{(n')^2+n'-2}.$$

To include the multiple node move in our algorithm, at each iteration of the loop in line 2 of Algorithm 1, we choose either the single node move or the multiple node move with probability $\frac{1}{2}$.

4 Simulations

We perform simulations to test the convergence and accuracy of our algorithm, which we call `segdup`. We first simulate a species tree S using a pure birth (Yule) process with rate 1, stopping when there are n_H species.

Given a species tree S, we simulate gene trees as follows:

- Start with a single gene copy for each of n_P gene trees at the root species of S.
- Each gene lineage may independently duplicate with rate r_B per lineage. When a gene lineage duplicates, each other lineage in the same species at the time may also duplicate, independently with probability p_J. This is considered as a single segmental duplication.
- At each speciation, with probability p_C each gene lineage will descend to one child species and be lost in the other (chosen independently at random); otherwise it descends to both child species.

We then input the resulting gene trees $\mathcal{G}$ into `segdup` with costs δ, λ, and run it for n iterations, to find a reconciliation between $\mathcal{G}$ and S. Additionally, we input the gene trees to MultRec (Dondi et al., 2019) to find an MPR between $\mathcal{G}$ and S. We then compute:

- The relative cost of the reconciliation found by `segdup` to that found by MultRec;
- The frequency of occurrences where `segdup` finds a better, equal, or worse cost than MultRec;
- The running time taken by `segdup` and MultRec (in seconds); and
- The relative cost of the reconciliation found by `segdup` to that of the true reconciliation.

Note that although the tool MultRec is supposed to find a guaranteed MPR, it does not do so in practice[2]. Thus, for practical purposes, it can also be considered as a heuristic that returns a valid, but not necessarily optimal, reconciliation.

[2]We have corresponded with the authors of MultRec; they have determined that their implementation is faulty, and a fixed implementation would take a much longer running time.

Parameter	Default	Minimum	Maximum
n_H	50	10	100
n_P	20	10	100
r_B	2	1	5
p_J	0.5	0.2	1
p_C	0.5		

Table 1: Parameter values for the simulations.

We vary each parameter (except for p_C, which is fixed at 0.5) for tree generation in turn, holding the others fixed at default values (shown in Table 1). For the reconciliation, we use $\delta = 10$, $\lambda = 1$, which is intended to present a challenging inference (the algorithm used by MultRec has an exponential running time with a base of $\lceil \frac{\delta}{\lambda} \rceil = 10$). We run 100 replicates for each parameter setting.

We show the results for varying duplication rate (r_B) for $n = 10^4$ and 10^5 iterations in Figure 3; the results for the other parameters are shown in Figures S1–S3. As each iteration can be performed quickly, even 10^5 is a relatively small number of iterations for a single instance.

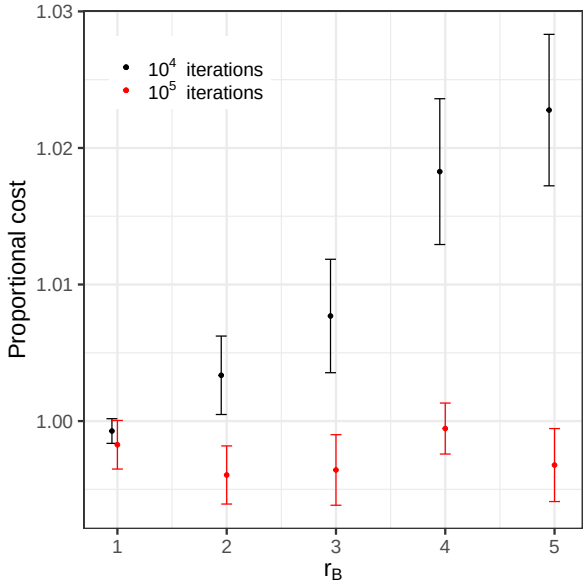
In Figure 3a, we see that for 10^5 iterations, `segdup` performs better than MultRec on average in all cases. Conversely, for 10^4 iterations, the performance of `segdup` degrades as the duplication rate increases, but even at the highest level the average cost of the reconciliation found by `segdup` is within 3% of that found by MultRec. It appears that for the larger instances, 10^4 is not always a sufficient number of iterations for the MCMC algorithm to properly converge; however, 10^5 is almost always sufficient.

For smaller duplication rates, the two methods agree for most cases (Figure 3b), suggesting that they have both found the optimal reconciliation. As the duplication rate increases, both `segdup` and MultRec fail to find the optimal solution more often; with 10^5 iterations, `segdup` usually finds the optimal reconciliation more often than MultRec, while for 10^4 iterations the situation is reversed.

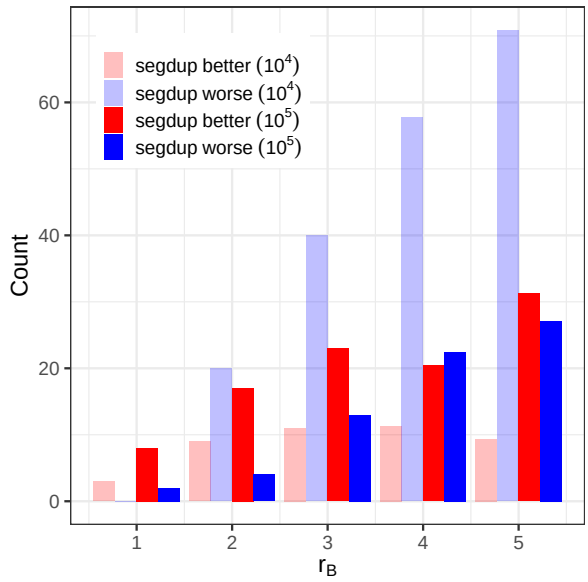
The running time of both `segdup` and MultRec grows with the duplication rate (Figure 3c). For small duplication rate, MultRec is faster than `segdup`, but as the duplication rate increases, `segdup` becomes faster, even for 10^5 iterations. This is consistent with the predicted performance of MultRec being exponential in the optimal number of segmental duplications.

Finally, in Figure 3d, we see that `segdup` consistently finds a reconciliation that is cheaper than the true reconciliation, as expected (since the true reconciliation is not necessarily optimal). However, the difference is relatively slight, and the reconciliation found by `segdup` is usually within 10% of the cost of the true reconciliation. The gap appears to narrow slightly as the duplication rate increases. As expected, running `segdup` for 10^5 iterations produces a better cost on average than for 10^4 iterations.

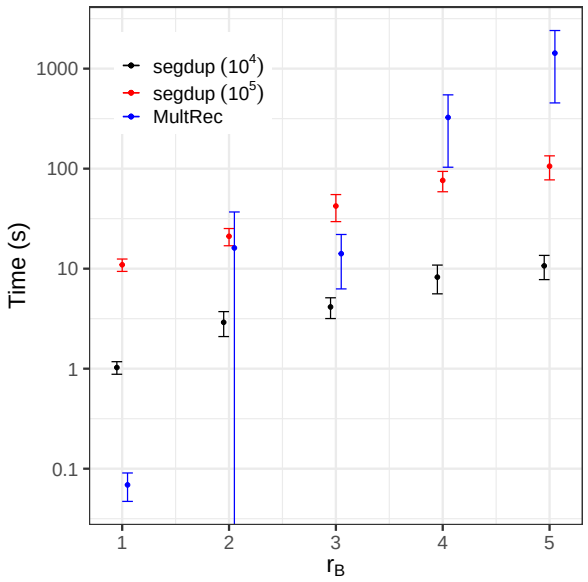
We see similar patterns when varying the other parameters. The performance of `segdup` appears relatively unaffected by the number of species (Figure S1), and decreases with the number of genes (Figure S2) and the probability of joint duplication (Figure S3). In all cases, `segdup` with 10^5 iterations always outperforms MultRec



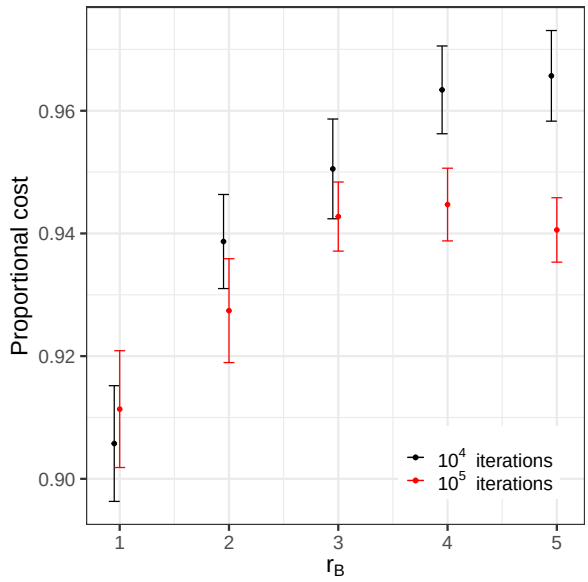
(a) Relative cost of `segdup` vs MultRec



(b) Frequencies of `segdup` vs MultRec



(c) Running time



(d) Relative cost of `segdup` vs true reconciliation

Figure 3: Comparison between `segdup` and MultRec and the true reconciliation. We vary r_B, the duplication rate, with 100 replicates for each parameter value. Error bars are twice standard errors.

on average.

The running time of `segdup` appears to grow approximately linearly with the number of species and number of genes, and in general exhibits a relatively smooth trend. Conversely, the running time of MultRec varies widely. Further investigation suggests that in most cases, MultRec solves the problem quickly, but when it does not, it can take a very long time, resulting in a large mean and standard error. This again appears to be a consequence of MultRec being exponential in the number of segmental duplications, which is difficult to determine *a priori* from a fixed problem size. In comparison, the time taken by `segdup` is more predictable as it runs for a fixed number of iterations.

In summary, our simulations show that `segdup`, while not always optimal, almost always outperforms MultRec with 10^5 iterations. It does so without the exponential running time of MultRec, which means that it is also faster for more complicated cases with many duplications.

5 Real data results

5.1 Eukaryote dataset

We re-analyse a dataset of 53 gene trees over a species tree of 16 eukaryotes (Guigó et al., 1996), shown in Figure 4. This dataset has been analysed several times (Page and Cotton, 2001; Bansal and Eulenstein, 2008; Dondi et al., 2019) for the presence of segmental duplications. Most recently, Dondi et al. (2019) showed that for a fixed loss cost of $\lambda = 1$ and a duplication cost of $\delta \in [28, 61]$, the inferred MPR has 5 segmental duplications. This is also the same reconciliation found by Paszek et al. (2020) using the so-called maximal path (MP) score. However, when $\delta > 61$, the inferred MPR has only 4 segmental duplications, with a duplication above the Tetrapoda clade (marked with a T in Figure 4) vanishing.

This range of values for δ may appear rather high. To put this in context, note that if the LCA mapping has k duplications assigned to the wrong species, `segdup` must reassign those k nodes, resulting in at least k more losses. Since k is positively related to the number of gene trees, the value of δ should be proportional to the number of input gene trees.

We concentrate on the case $\delta = 50$, for which the inferred MPR has 5 segmental duplications. In Figure 5, we show the convergence of `segdup` for one run over 10^5 iterations from an initial temperature of 20 (this takes less than a minute to run on a desktop computer). It is clear that even with this relatively small number of iterations, we converge well to the target distribution. At high temperatures, we obtain a relatively stable cost and around 7–10 segmental duplications (the LCA mapping has 9 segmental duplications). As the temperature decreases to 0, the cost slowly decreases and the number of segmental duplications decreases to the optimal value of 5.

We run `segdup` 20 times, each for 10^5 iterations on this instance. For 5 out of those 20 times, the algorithm converges to the MPR inferred by MultRec, which contains 5 (segmental) duplications and 468 losses. For 13 out of

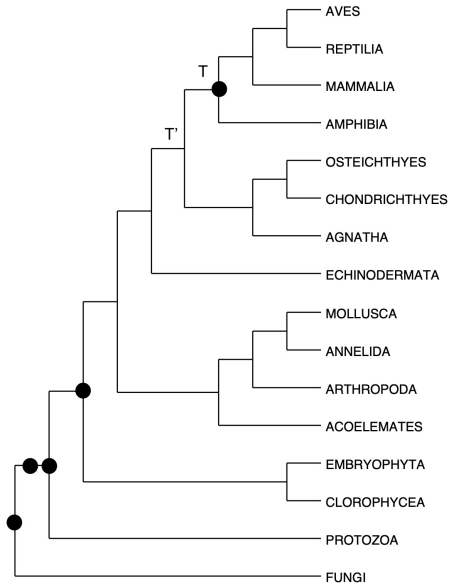
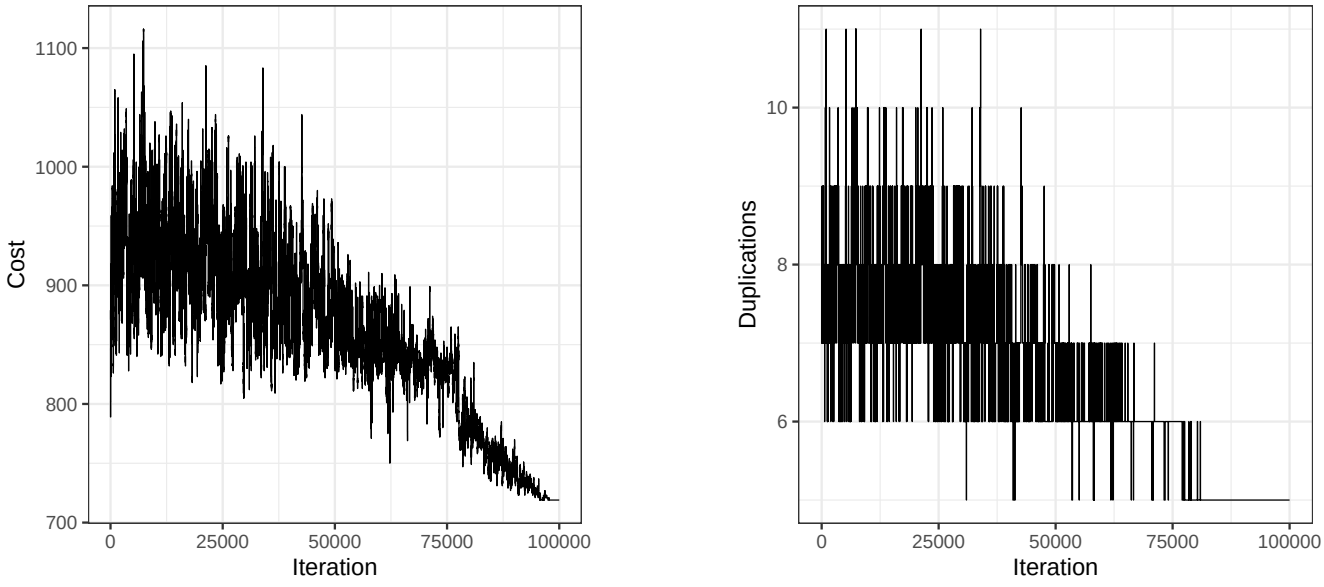


Figure 4: The species tree from Guigó et al. (1996). The Tetrapoda clade is labelled with a T, and the segmental duplications in the inferred MPR for $\delta = 50$ are indicated by black circles.



(a) Cost vs number of iterations

(b) Number of duplications vs number of iterations

Figure 5: Trace plots for one run of `segdup` for the eukaryote dataset for 10^5 iterations.

the other 15 times, we find a near-optimal reconciliation with 5 duplications and 469 losses, which differs from the MPR only in that the duplication above the Tetrapoda clade is now moved one species branch higher (to the branch marked T' in Figure 4). Although these two solutions are very close in terms of cost, they differ significantly in the placements of the individual gene duplications, so they are not easily obtainable from each other — for example, they cannot be derived from each other by a single move of either type in our algorithm.

This illustrates both a strength and a weakness of our algorithm. The weakness is that it does not guarantee optimality. On the other hand, the strength is that it always returns a valid reconciliation, which (given convergence) is usually optimal or very close to it. By running `segdup` several times, we can recover multiple solutions that may give more biological insight than a single MPR.

To explore this further, we examine the relative frequencies of these two duplications at different temperatures. For a range of final temperatures between 0 and 10, we run `segdup` from an initial temperature of 20, linearly decreasing over 10^5 iterations to the final temperature. We then hold the temperature fixed at the final temperature and run for another 10^5 iterations, recording every 10th reconciliation. Finally, we determine how many of these reconciliations have a duplication above the Tetrapoda clade, and how many have a duplication at the T' branch.

The results are shown in Figure 6. At high final temperature, we see that the duplication at the T' branch occurs slightly more often (about 70%) than above the Tetrapoda clade; this varies more as the temperature decreases, around a similar number. This indicates that even though the optimal solution places the duplication at the T branch, when a Boltzmann distribution is imposed the overall space of reconciliations actually supports a duplication at the T' branch more, since it occurs slightly more often at non-zero temperatures. As the final temperature decreases, the chain of reconciliations tends to collapse to one of the two solutions, as expected, but again with a slight preference for the duplication at the higher branch.

Our results support the findings of Dondi et al. (2019), who considered that the 4-duplication solution for $\delta > 61$ shed doubt on the existence of the duplication above the Tetrapoda clade. Our results show that even at lower δ, a holistic view of the Boltzmann distribution on the reconciliation space also sheds doubt on the location of this duplication. This suggests that studying the Boltzmann distribution can provide further biological insight above merely considering most parsimonious reconciliations.

5.2 Yeast dataset

We additionally analyse a dataset of 2379 gene trees over a 16-taxon species tree of yeast species (Butler et al., 2009). We use the values $\delta = 1700, \lambda = 1$. In Figure 8, we show the convergence of `segdup` for one run over 10^5 iterations from an initial temperature of 2000 (chosen manually). This took about 3 hours on the same desktop computer.

This is a much larger dataset than the eukaryote dataset, and it is clear that the algorithm does not converge very well for this number of iterations. Nevertheless, `segdup` is still able to find a solution that is much better than

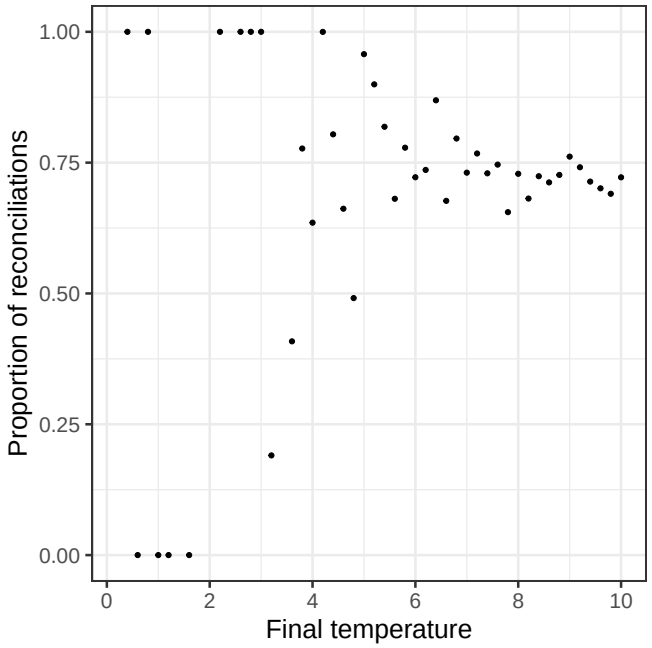


Figure 6: The proportion of sampled reconciliations with a duplication at the higher branch vs above the Tetrapoda clade, for varying final temperature.

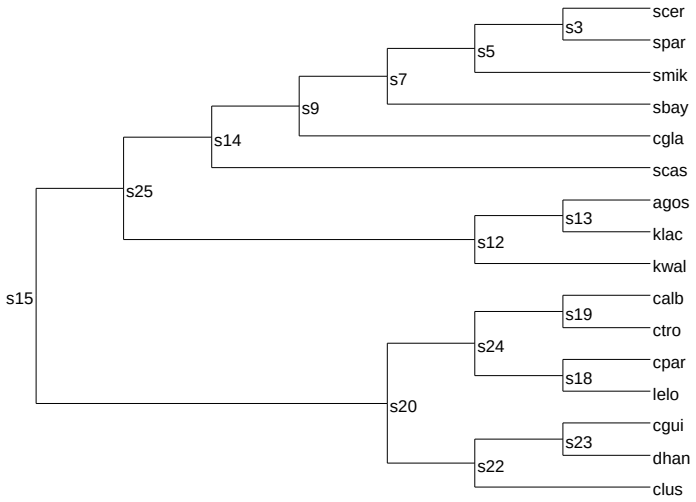
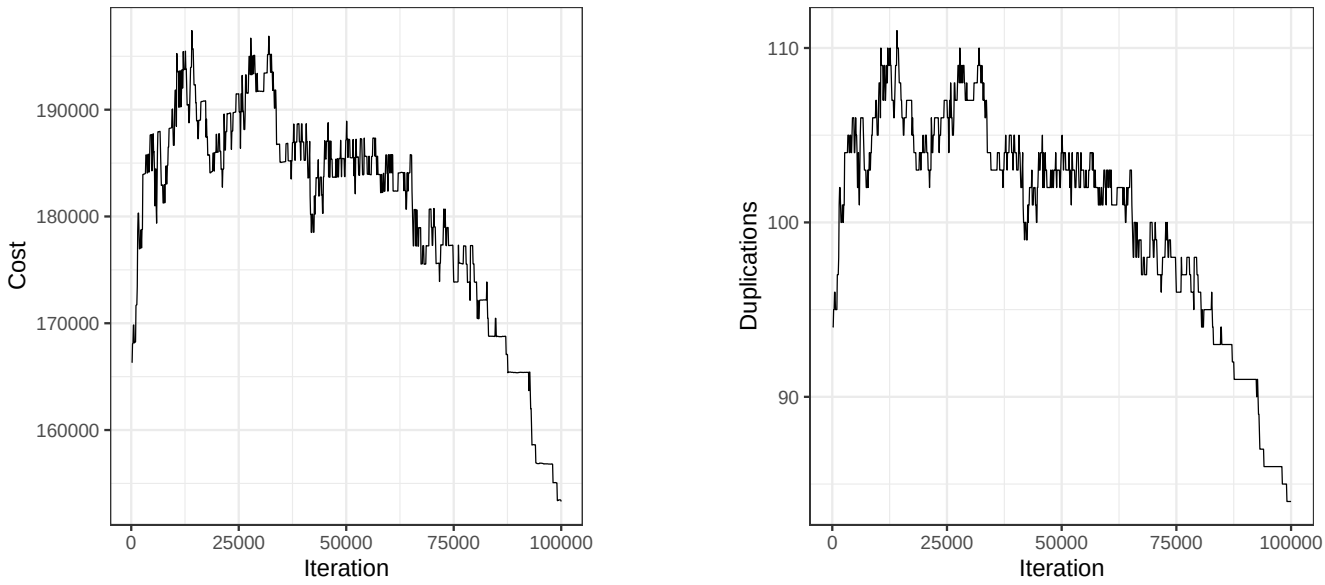


Figure 7: The species tree for the yeast dataset (Butler et al., 2009). Full species names are given in Table S1.



(a) Cost vs number of iterations

(b) Number of duplications vs number of iterations

Figure 8: Trace plots for one run of `segdup` for the yeast dataset for 10^5 iterations.

the LCA mapping, with a cost of 153,308 (84 segmental duplications and 10,508 losses) vs 169,574 (96 segmental duplications and 6,374 losses) for the LCA mapping.

In Table S2, we show the number of individual duplications inferred at each internal species vertex. Our inferred reconciliation supports the whole-genome duplications proposed in Marcet-Houben and Gabaldón (2015) at vertices s14 and s25, with 471 and 445 duplications respectively. However, there is even stronger support for large-scale duplications at vertices s12 and s15 (the root species), with 630 and 1253 duplications inferred respectively. This illustrates the tendency of duplications to 'float' (be placed higher in the species tree) when the chain is not fully converged.

While the lack of convergence of the algorithm means that these results may not be entirely definitive, this analysis nevertheless demonstrates the potential of this approach to produce useful results on large datasets even without a guarantee of optimality.

6 Discussion

In this paper, we have devised a probabilistic algorithm to solve the most parsimonious reconciliation problem for segmental duplications. Our algorithm works by imposing a Boltzmann distribution on the space of reconciliations, then using a Gibbs sampling-like MCMC combined with simulated annealing to reach an MPR. We show that our algorithm performs well on both simulated data and real datasets that have been scrutinised several times before for the presence of segmental duplications.

The ideas underlying our algorithm lay the foundations for avenues of research in several different directions.

First, although we use the Boltzmann distribution here as a means to reach an MPR, analysing the Boltzmann distribution on the space of reconciliations at fixed temperatures can give insight into the structure of the reconciliation space, as we show for the eukaryote dataset on a limited scale. A common criticism of discrete methods in phylogenetics is that they often give a single 'point estimate', with little quantification of the uncertainty of the solution. Some methods propose to address this issue by analysing multiple solutions; using a Boltzmann weighting for these solutions allows for a more nuanced analysis than, e.g., simply considering all optimal solutions, for which there are known to be an exponential number anyway. Our algorithm allows us to construct a sample from the Boltzmann distribution without needing the model to be solvable by dynamic programming, and is therefore useful on a much larger class of models.

The basic framework of our algorithm is essentially model-free; while it performs well on the segmental duplications model, it can be easily adapted to any parsimony reconciliation model. The solutions of such models have been largely limited to those amenable to dynamic programming; as such, it has so far only been possible to solve models that contain only a small number of (types of) evolutionary processes. This has resulted in a variety of solvable but incomplete models, while a larger model unifying many processes would be more realistic.

One example of such a model is in Duchemin (2017), who formulated a model with a combined score involving tree scores from sequences, individual duplication-loss events, synteny events, and segmental events. Although a heroic effort, this model has been completely intractable by standard solution methods up to now. Our work represents a new avenue to approach such complex unified models, as it only requires the ability to calculate the cost of a reconciliation, which is generally much easier than a calculation of likelihood under a statistical model.

In particular, it is well known that a number of even basic reconciliation problems (e.g., undated DTL reconciliation) are NP-hard, and so any kind of efficient algorithm that can guarantee finding an MPR must be non-polynomial (and likely exponential) in time. Our algorithm circumvents this by not guaranteeing an MPR, but because it always returns a valid reconciliation, it can also be thought of as a (very advanced) heuristic that can potentially be used as a practical solution for these difficult problems. Our work is thus a framework that can be used to make progress on difficult reconciliation problems, extending both their realism and their practical usefulness.

Data Availability

The code for the simulations and real data analysis can be found at `https://github.com/mcharleston/segdup` for `segdup` and `https://github.com/mcharleston/kowhai` for the simulator.

Acknowledgments

The authors thank Manuel Lafond and Reza Kalhor for useful discussions.

Author(s') disclosure

The authors declare that they have no conflicts of interest.

Funding statement

This work has been partially funded by French Agence Nationale de la Recherche through the CoCoAlSeq Project (ANR-19-CE45-0012).

References

Anselmetti, Y., Delabre, M., and El-Mabrouk, N. Reconciliation with segmental duplication, transfer, loss and gain. In *RECOMB International Workshop on Comparative Genomics*, pages 124–145. Springer, 2022.

Bansal, M. S. and Eulenstein, O. The multiple gene duplication problem revisited. *Bioinformatics*, 24(13):i132–i138, 2008.

Butler, G., Rasmussen, M. D., Lin, M. F., Santos, M. A., Sakthikumar, S., Munro, C. A., Rheinbay, E., Grabherr, M., Forche, A., Reedy, J. L., et al. Evolution of pathogenicity and sexual reproduction in eight *Candida* genomes. *Nature*, 459(7247):657–662, 2009.

Chan, Y., Ranwez, V., and Scornavacca, C. Reconciliation-based detection of co-evolving gene families. *BMC Bioinform.*, 14(1):332–340, 2013.

Chan, Y. and Robin, C. Reconciliation of a gene network and species tree. *J. Theor. Biol.*, 472:54–66, 2019.

Chan, Y., Ranwez, V., and Scornavacca, C. Inferring incomplete lineage sorting, duplications, transfers and losses with reconciliations. *J. Theor. Biol.*, 432:1–13, 2017.

Chauve, C., Ponty, Y., and Zanetti, J. P. P. Evolution of genes neighborhood within reconciled phylogenies: an ensemble approach. *BMC Bioinform.*, 16:1–9, 2015.

Delabre, M., El-Mabrouk, N., Huber, K. T., Lafond, M., Moulton, V., Noutahi, E., and Castellanos, M. S. Evolution through segmental duplications and losses: a super-reconciliation approach. *Alg. Mol. Biol.*, 15:1–15, 2020.

Dondi, R., Lafond, M., and Scornavacca, C. Reconciling multiple genes trees via segmental duplications and losses. *Alg. Mol. Biol.*, 14(1):7, 2019.

Doyon, J.-P., Ranwez, V., Daubin, V., and Berry, V. Models, algorithms and programs for phylogeny reconciliation. *Brief. Bioinform.*, 12(5):392–400, 2011.

Drinkwater, B., Qiao, A., and Charleston, M. A. WiSPA: A new approach for dealing with widespread parasitism. *arXiv preprint arXiv:1603.09415*, 2016.

Duchemin, W. *Phylogeny of dependencies and dependencies of phylogenies in genes and genomes.* PhD thesis, Université de Lyon, 2017.

Goodman, M., Czelusniak, J., Moore, G. W., Romero-Herrera, A. E., and Matsuda, G. 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.

Górecki, P. and Tiuryn, J. DLS-trees: a model of evolutionary scenarios. *Theoret. Comput. Sci.*, 359(1-3):378–399, 2006.

Guigó, R., Muchnik, I., and Smith, T. F. Reconstruction of ancient molecular phylogeny. *Mol. Phyl. Evol.*, 6(2): 189–213, 1996.

Hasić, D. and Tannier, E. Gene tree species tree reconciliation with gene conversion. *J. Math. Biol.*, 78(6):1981–2014, 2019a.

Hasić, D. and Tannier, E. Gene tree reconciliation including transfers with replacement is NP-hard and FPT. *J. Comb. Optim.*, 38(2):502–544, 2019b.

Li, Q., Scornavacca, C., Galtier, N., and Chan, Y. The multilocus multispecies coalescent: a flexible new model of gene family evolution. *Syst. Biol.*, 70(4):822–837, 2021.

Marcet-Houben, M. and Gabaldón, T. Beyond the whole-genome duplication: phylogenetic evidence for an ancient interspecies hybridization in the baker's yeast lineage. *PLoS Biol.*, 13(8):e1002220, 2015.

Menet, H., Daubin, V., and Tannier, E. Phylogenetic reconciliation. *PLoS Comp. Biol.*, 18(11):e1010621, 2022.

Page, R. D. and Cotton, J. Vertebrate phylogenomics: reconciled trees and gene duplications. In *Biocomputing 2002*, pages 536–547. World Scientific, 2001.

Paszek, J., Tiuryn, J., and Górecki, P. Minimizing genomic duplication episodes. *Comput. Biol. Chem.*, 89:107260, 2020.

Paszek, J. and Górecki, P. Efficient algorithms for genomic duplication models. *IEEE/ACM Trans. Comput. Biol. Bioinform.*, 15(5):1515–1524, 2018. doi: 10.1109/TCBB.2017.2706679.

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

Wu, Y.-C., Rasmussen, M. D., Bansal, M. S., and Kellis, M. Most parsimonious reconciliation in the presence of gene duplication, loss, and deep coalescence using labeled coalescent trees. *Genome Res.*, 24(3):475–486, 2014.

Zhang, L. On a Mirkin-Muchnik-Smith conjecture for comparing molecular phylogenies. *J. Comp. Biol.*, 4(2): 177–187, 1997.

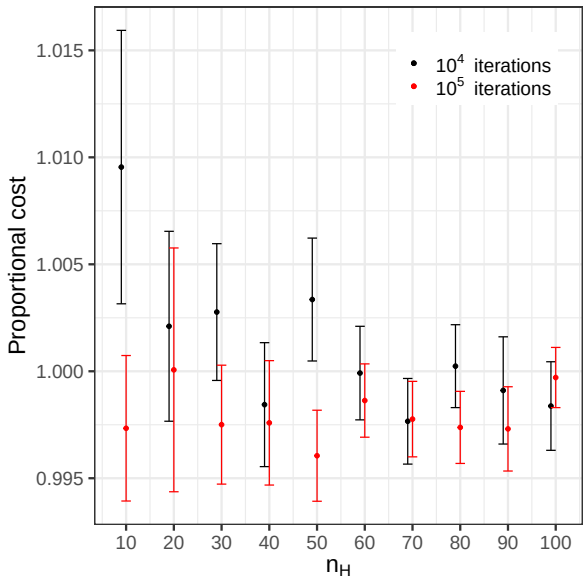
A Supplementary Figures and Tables

Abbreviation	Full name
scer	*Saccharomyces cerevisiae*
spar	*Saccharomyces paradoxus*
smik	*Saccharomyces mikatae*
sbay	*Saccharomyces bayanus*
cgla	*Candida glabrata*
scas	*Saccharomyces castelli*
agos	*Ashbya gossypii*
klac	*Kluyveromyces lactis*
kwal	*Kluyveromyces waltii*
calb	*Candida albicans*
ctro	*Candida tropicalis*
cpar	*Candida parapsilosis*
lelo	*Lodderomyces elongisporus*
cgui	*Candida guilliermondii*
dhan	*Debaryomyces hansenii*
clus	*Candida lusitaniae*

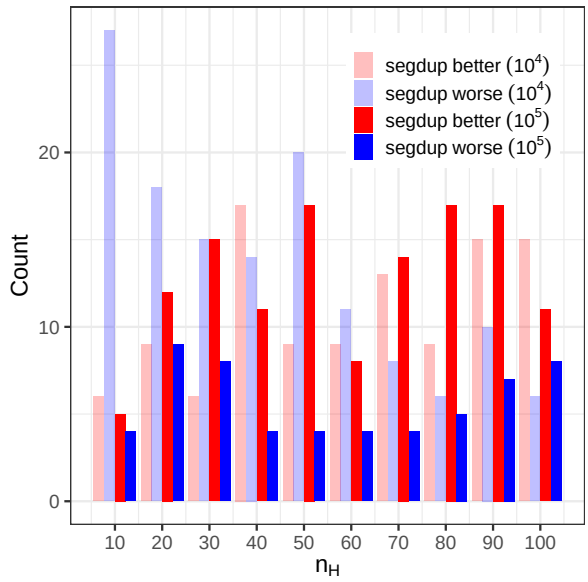
Table S1: Species name abbreviations for the yeast dataset.

Species	Number of duplications
s3	13
s5	45
s7	169
s9	137
s12	630
s13	5
s14	471
s15	1253
s18	69
s19	73
s20	396
s22	373
s23	19
s24	82
s25	445

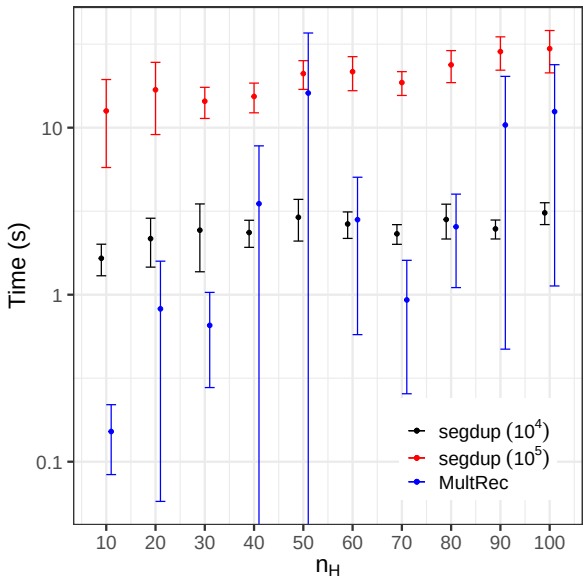
Table S2: Number of individual duplications found at each species vertex for the yeast dataset.



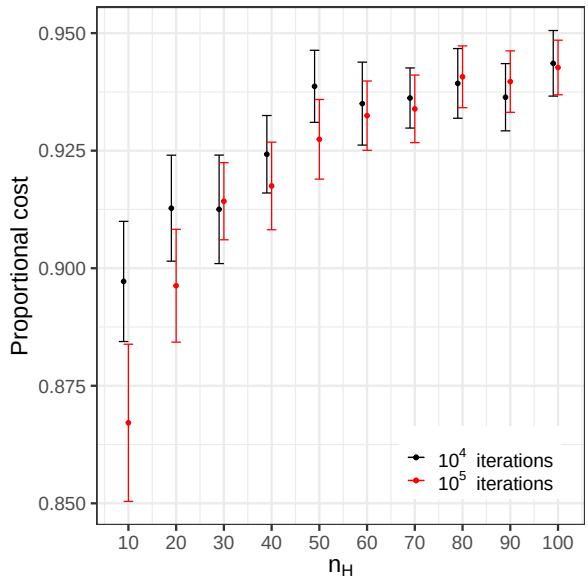
(a) Relative cost of `segdup` vs MultRec



(b) Frequencies of `segdup` vs MultRec

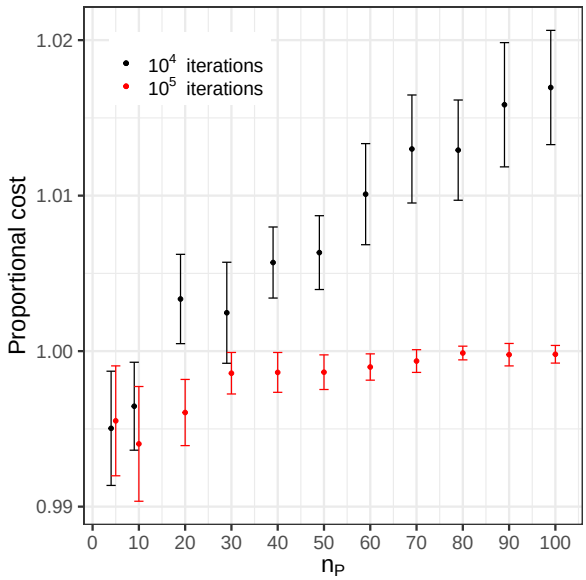


(c) Running time

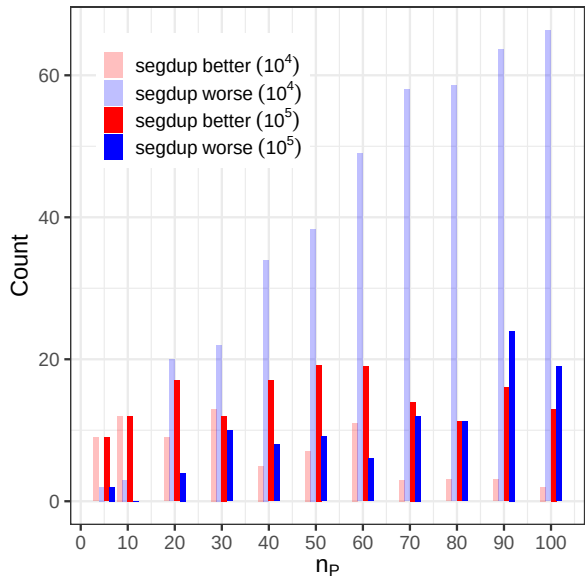


(d) Relative cost of `segdup` vs true reconciliation

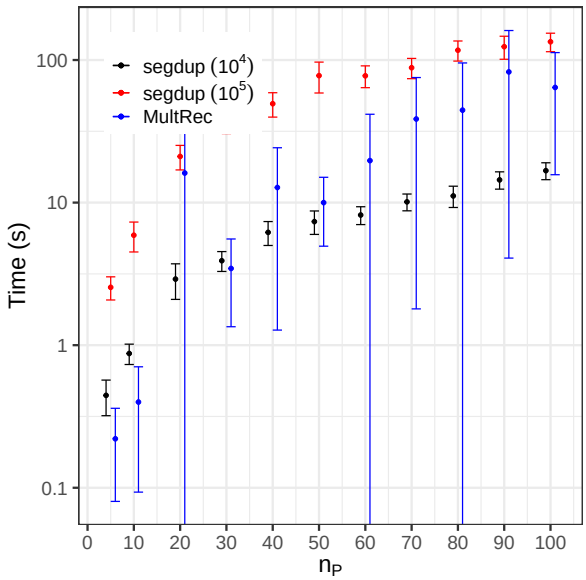
Figure S1: Comparison between `segdup` and MultRec and the true reconciliation. We vary n_H, the number of species, with 100 replicates for each parameter value. Error bars are twice standard errors.



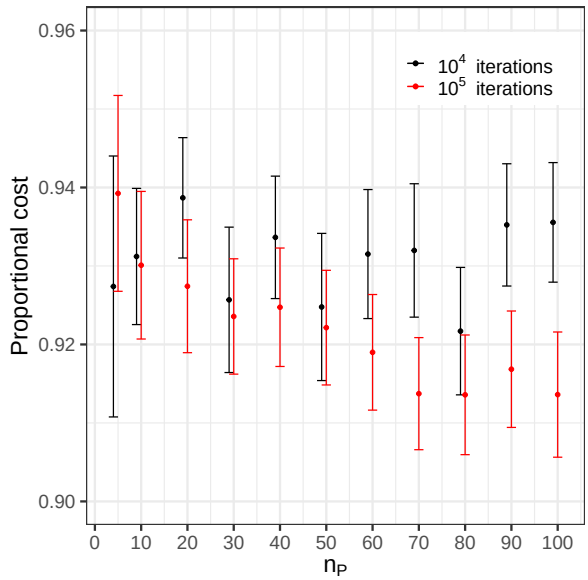
(a) Relative cost of `segdup` vs MultRec



(b) Frequencies of `segdup` vs MultRec

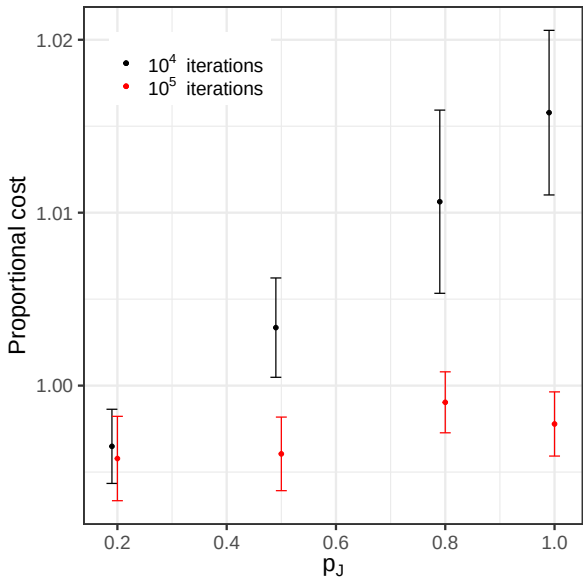


(c) Running time

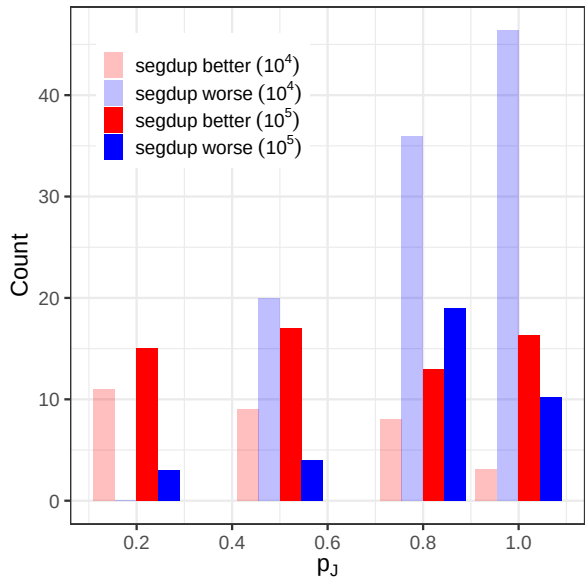


(d) Relative cost of `segdup` vs true reconciliation

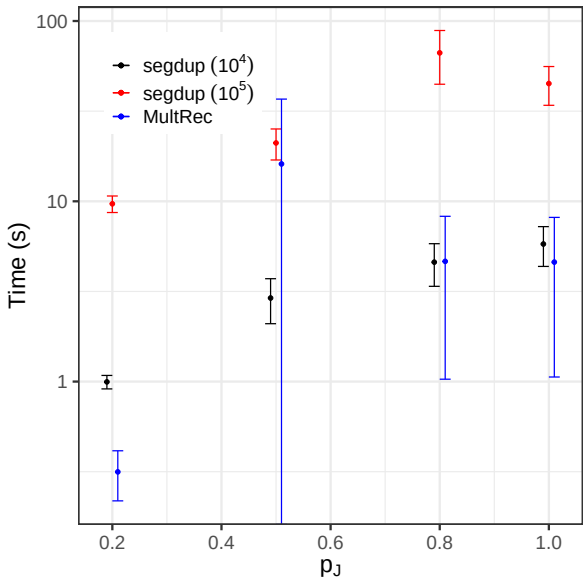
Figure S2: Comparison between `segdup` and MultRec and the true reconciliation. We vary n_P, the number of gene trees, with 100 replicates for each parameter value. Error bars are twice standard errors.



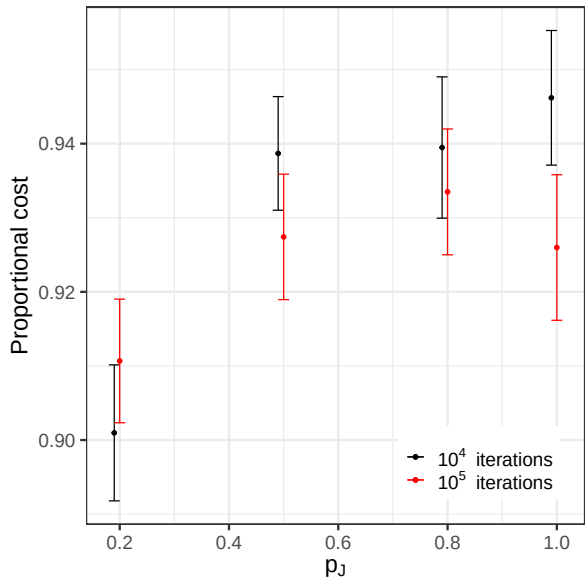
(a) Relative cost of `segdup` vs MultRec



(b) Frequencies of `segdup` vs MultRec



(c) Running time



(d) Relative cost of `segdup` vs true reconciliation

Figure S3: Comparison between `segdup` and MultRec and the true reconciliation. We vary p_J, the probability of joint duplication, with 100 replicates for each parameter value. Error bars are twice standard errors.