Inferring Species Trees from Incongruent Multi-Copy Gene Trees Using the Robinson-Foulds Distance

Chaudhary et al., *Algorithms Mol. Biol.*, 2013

Journal club presentation by Ananya Yammanuru
June 27, 2018

Overall goal

Estimate a species tree from phylogenomic datasets

Challenges

- Gene tree incongruence
- Multi-copy gene grees

Gene Tree Incongruence

Means that gene trees have different topologies from each other and from the species tree.

- Could be due to evolutionary events
  - Recombination
  - Gene Duplication
  - Gene Loss
  - Deep Coalescence
  - Lateral Gene Transfer (LGT)
- Could also be due to gene tree estimation error

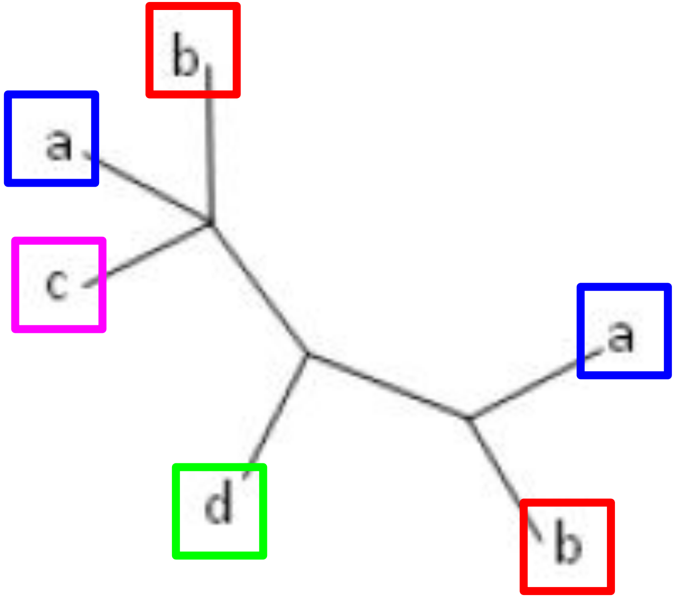
Gene Tree Heterogeneity / Discord / Incongruence

Means that gene trees have different topologies from each other and from the species tree.

- Could be due to evolutionary events
  - Recombination
  - **Gene Duplication**
  - **Gene Loss**
  - Deep Coalescence
  - Lateral Gene Transfer (LGT)
- Could also be due to gene tree estimation error

Gene Duplication and Multi-Copy Gene Trees

- **Multiple leaves** in a gene tree can have the **same species label**.
- Multi-labeled trees are called **mul-trees**.



Gene Loss

- Gene trees can be **missing species** that appear in other gene trees.
- Each gene tree in the collection of gene trees can be labeled with a different set of species labels.

Note that missing data can be due to other causes that true gene loss, for example, data collection and sampling errors.

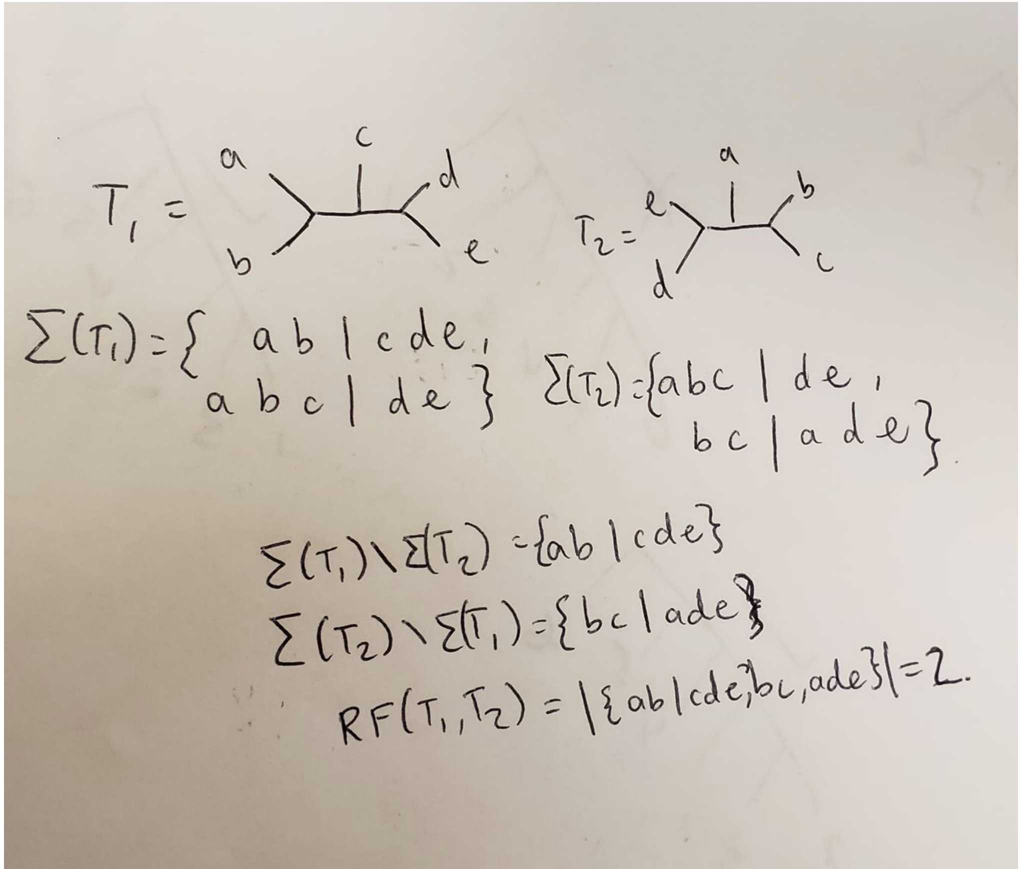
Robinson-Foulds (RF) Distance

Let T1 and T2 be unrooted, binary, singly-labeled trees. Then RF distance is defined

$$RF(T1, T2) := |\ (C(T1) \setminus C(T2))\ U\ (C(T2) \setminus C(T1))\ |$$

where C(T) is the set of bipartitions (splits) induced by the internal edges of tree T

RF Distance Example


$T_1 =$
a
b
c
d
e
$T_2 =$
e
d
a
b
c

$\Sigma(T_1) = \{ ab \mid cde, abc \mid de \}$ $\Sigma(T_2) = \{abc \mid de, bc \mid ade\}$.

$\Sigma(T_1) \setminus \Sigma(T_2) = \{ab \mid cde\}$

$\Sigma(T_2) \setminus \Sigma(T_1) = \{bc \mid ade\}$

$RF(T_1, T_2) = |\{ab \mid cde, bc, ade\}| = 2.$

RF distances for mul-trees

- Computing the RF distance between two mul-trees is NP-hard (Theorem 1 proved by reduction to Exact Cover by 3-Sets).
- BUT computing the RF distance between a mul-tree and a supertree (singly-labeled) tree can be done in polynomial time by using the “extended supertree”. This is the basis of most of the remaining RF calculations.

Mul-tree and its extended supertree

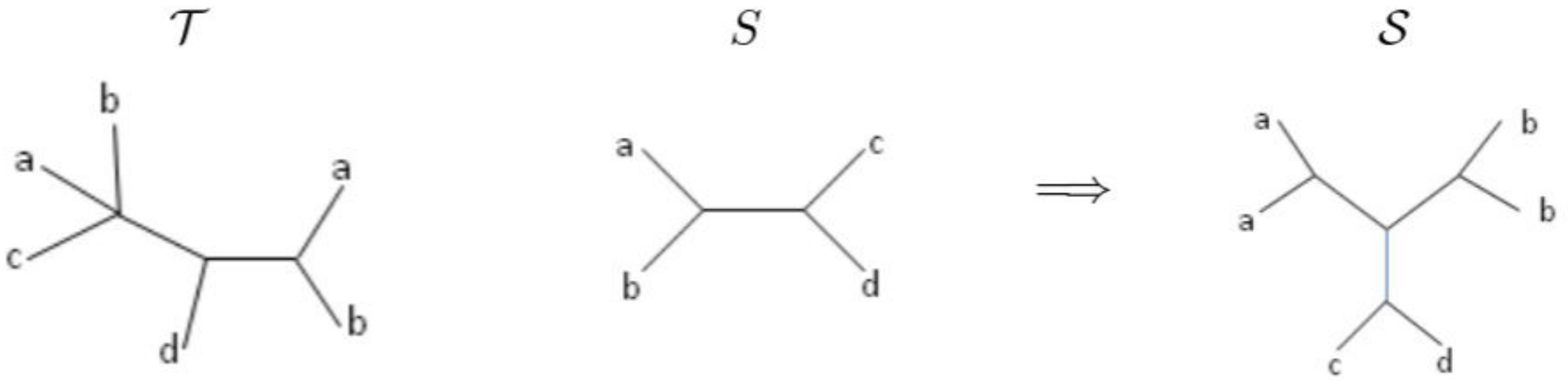
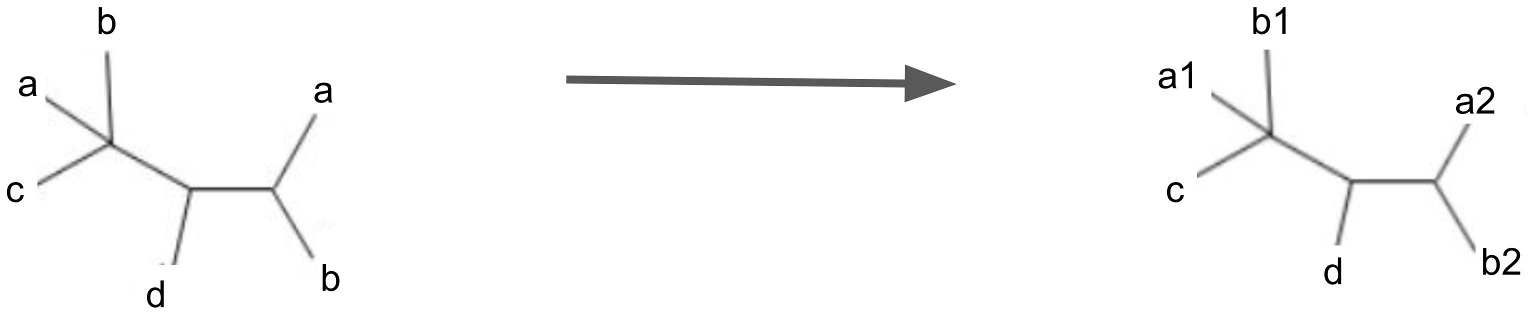


Fig. 3. Input mul-trees $\mathcal{T}$ and the supertree S. The extended supertree $\mathcal{S}$ is also shown.

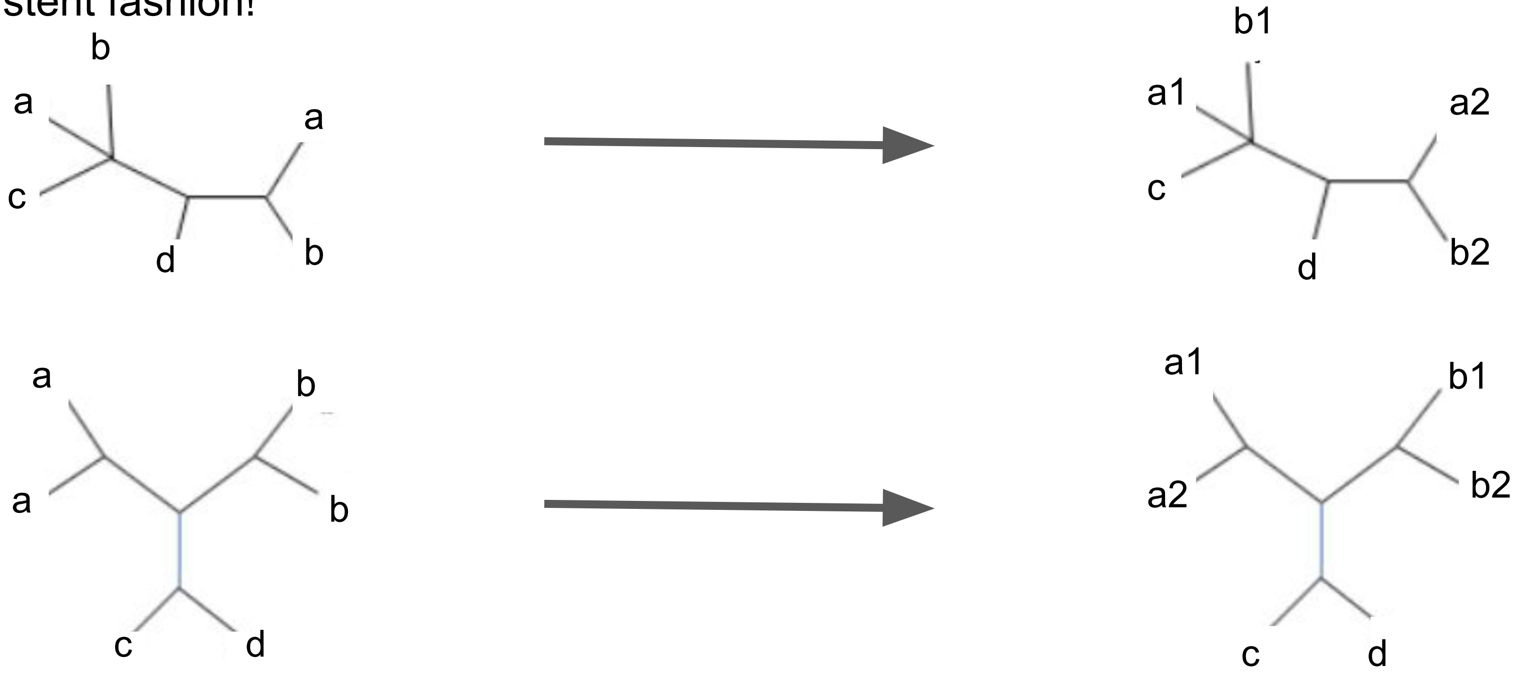
Full differentiation

Basically, differentiate between labels that appear multiple times!



Mutually consistent full differentiation

Basically, differentiate labels that appear multiple times in the two mul-trees in a consistent fashion!



Theorem 3

Let T be an mul-tree and let S be a supertree (singly-labeled) tree. Then all mutually consistent full differentiations of T and the extended supertree for S give the same RF distance.

Proof: *Basically show changing the label order does not change the RF distance.*

For each label *l*, consider all the ways of relabeling T. If all the leaves with *l*-labels are on the same side of the split, then the same split exists in S. If the *l*-labeled leaves are on different sides of the split, then no matter what labels they're given, the split does not exist in S.

Cool conclusion: You only need to calculate RF distance for only one full differentiation of the mul-tree.

RF Supertree for Mul-trees (MulRF) Problem

- **Input:** A profile of *P* = (*T1*, *T2*, …, *Tk*) of unrooted mul-trees
- **Output:** A binary supertree *T** for *P* that minimizes the sum of the RF distance between *T** and each mul-tree Ti in *P*.

The MulRF problem (like the RF supertree problem) is an **NP-hard optimization problem**!

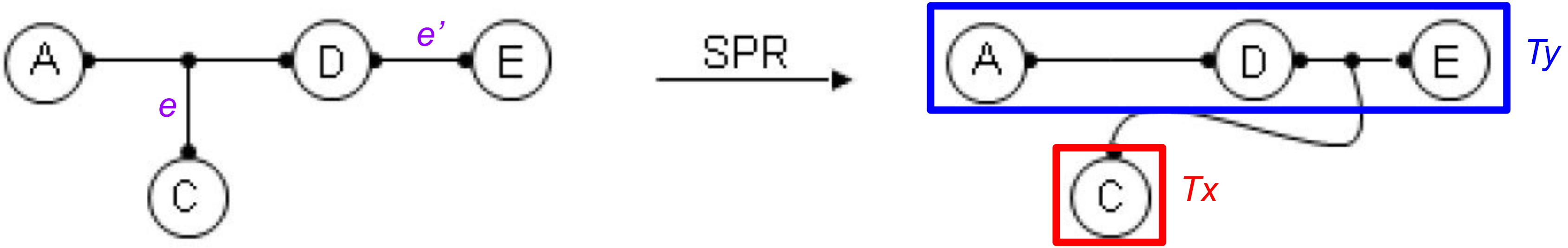
Overview of MulRF Heuristic

1. Compute an initial supertree *S* for a profile *P*.
2. Find each supertree S' in the **SPR neighborhood** of S, and score S' by computing RF(*P*, S').
    - Can be done in O(kn^2) time where k is the number of trees in P and n is the number of species.
3. Return the supertree *S'* for P in the SPR neighborhood of *S* with the best score.

Note: The choice of *S* seems likely to be really important!

Subtree Prune and Regraft (SPR) Neighborhood

1. **Prune:** Cut edge *e = (x,y)* in *T* to get to subtrees *Tx* and *Ty*.
2. **Regraft:** Regraft *Tx* to any edge *e'* in *Ty* by creating a new vertex in the middle of the edge *e'*.



Because there are O(n) choices for edge *e* and O(n) choices for edge e', there are O(n^2) trees in the **SPR neighborhood** of *T*.

Computing the RF distance between trees T and S

Let T and S be unrooted, binary, singly-labeled trees (“regular trees”).

Then you can compute the RF distance in linear time as follows.

1. Root T and S at an arbitrary shared leaf.
2. Compute the LCA for all nodes in T in O(n) time.
3. Compute the LCA mapping from S to T in O(n) time.
4. Compute the RF distance between S and T in O(n) time [Lemmas 2 and 3].

Least Common Ancestor (LCA) mapping between two trees

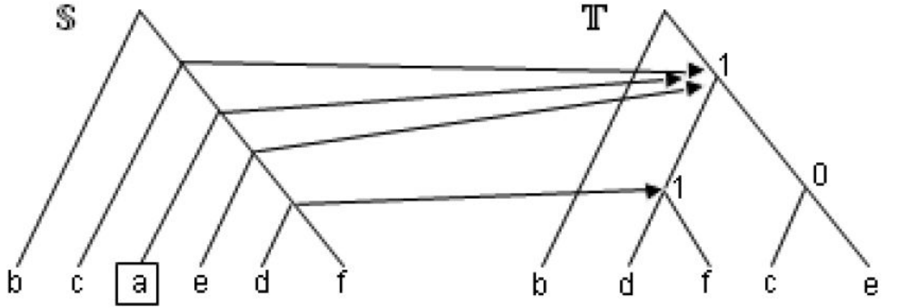


Fig. 6. The LCA mapping from $\mathbb{S}$ to $\mathbb{T}$. Vertex a in $\mathbb{S}$ is mapped to *null* as $a \notin \mathcal{L}(\mathbb{T})$. The internal vertices of $\mathbb{T}$ are labeled with the values of the vertex function.

This calculation can be done in O(n) time, using the bottom-up (DP?) algorithm specified in section 4 of [5].

Computing the RF distance between T and all trees in the SPR neighborhood of S

Suppose you already computed the RF distance between T and S.

But now you want to compute the RF distance between T and S’, where S’ is in the SPR neighborhood of S.

If you have selected S’ based on a specific ordering of SPR moves, then you can do this computation in **constant time** [Lemma 4].

Thus, the total time to compute the distance between T and every tree in the SPR neighborhood of S is O(n^2) time.

“Nice” Order for SPR Moves Example

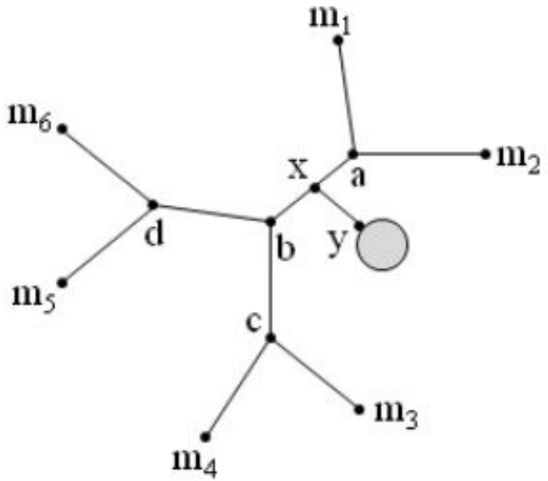


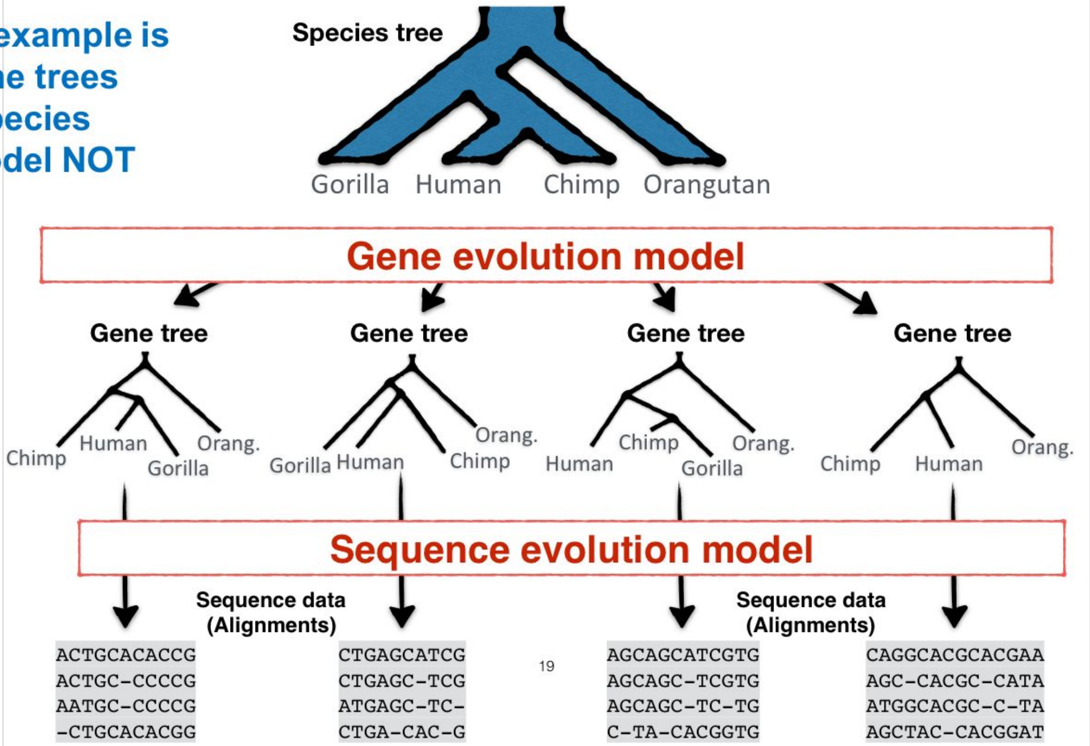
Fig. 7. A tree with a subtree regrafted at edge $\{a, b\}$. One iteration of vertices in the tree is $m_1, a, m_2, \quad a, b, c, m_3, c, m_4, c, b, d, m_5, d, m_6, d, b, a, m_1$. The resulting ordering $\aleph$ is $\{m_1, a\}, \{a, m_2\}, ..., \{a, m_1\}$.

Simulation Study

1. Simulate species trees under the Yule process using Mesquite (20 replicates for each model condition)
   - 50 species with height of 220 thousand years (tyrs)
   - 100 species with height of 440 tyrs
2. Simulate gene trees from species tree.
   - 150 gene trees for 50-species dataset
   - 300 gene trees for 100 species dataset
3. Delete 25% of leaves in gene trees to create missing data.
4. For each gene tree, a DNA multiple sequence alignment was simulated under the GTR+GAMMA+I model using Seq-Gen.

Steps 1-3 (taken from Siavash Mirarab)

Note that this example is simulating gene trees under multi-species coalescent model NOT GDL model!



```
ACTGCACACCG
ACTGC-CCCCG
AATGC-CCCCG
-CTGCACACGG
```

```
CTGAGCATCG
CTGAGC-TCG
ATGAGC-TC-
CTGA-CAC-G
```

```
AGCAGCATCGTG
AGCAGC-TCGTG
AGCAGC-TC-TG
C-TA-CACGGTG
```

```
CAGGCACGCACGAA
AGC-CACGC-CATA
ATGGCACGC-C-TA
AGCTAC-CACGGAT
```

Simulation Study, cont.

Gene trees were simulated under the following four scenarios:

- ***none***: No duplications, loss, or LGT events
  - Gene tree topologies differ from species tree due to estimation error only
- ***dl***: Simulate duplication and loss events using Arvestad et al.’s model with a rate of 0.002 events/gene per tyrs
- ***lgt***: Simulate LGT rate of 2 events/gene
- ***both***: Simulate duplication and loss events and then simulate LGT events (same rates as above)

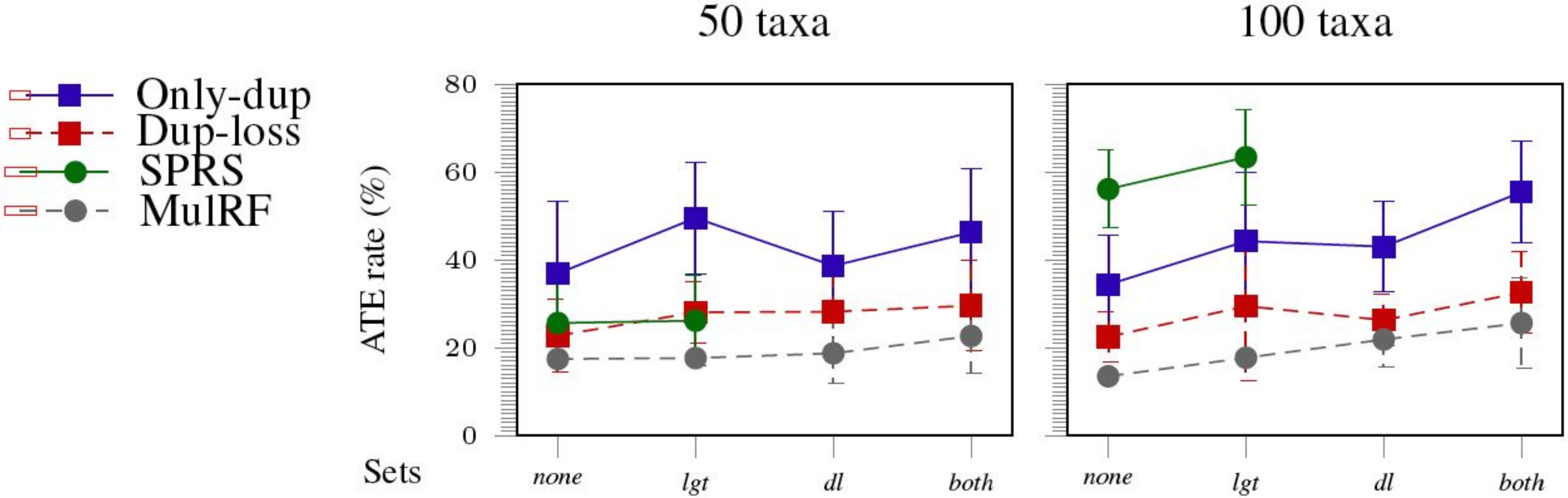
Species Tree Estimation

1. Gene trees were estimated using RAxML.
2. Root estimated gene trees at the midpoint for some Gene Tree Parsimony (GTP) methods.
3. Species trees were estimated from the estimated gene trees four different methods:
    - GTP for minimizing duplication events (dup-only)
    - GTP for minimizing duplication and loss events (dup-loss)
    - GTP for minimizing LGT events (SPR supertree; SPRS)
    - MulRF

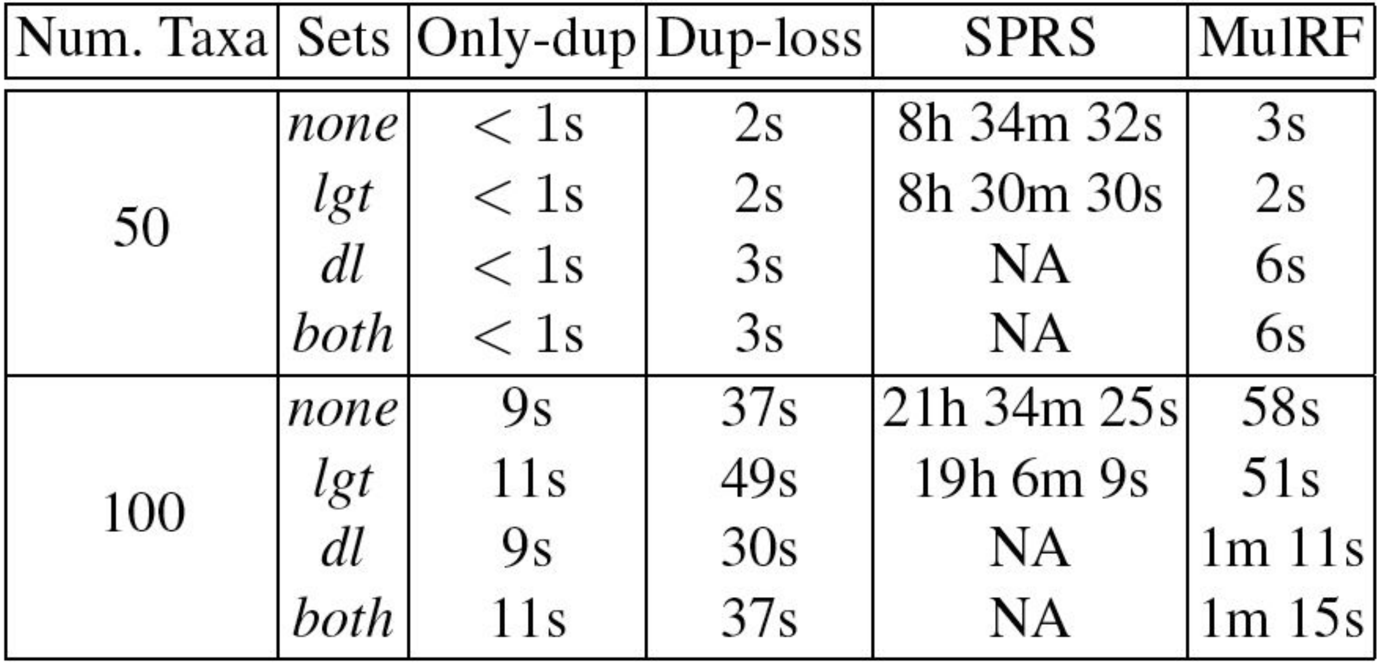
Species Tree Method Evaluation

- Error (Average Topological Error; ATE)
  - Normalized RF distance between the true species tree and the estimated species tree, averaged across all 20 replicate datasets for each model condition
- Running time
- Estimated number of duplication or loss events compared to the true number of duplication and loss events

Results - Species Tree Error (Figure 9)

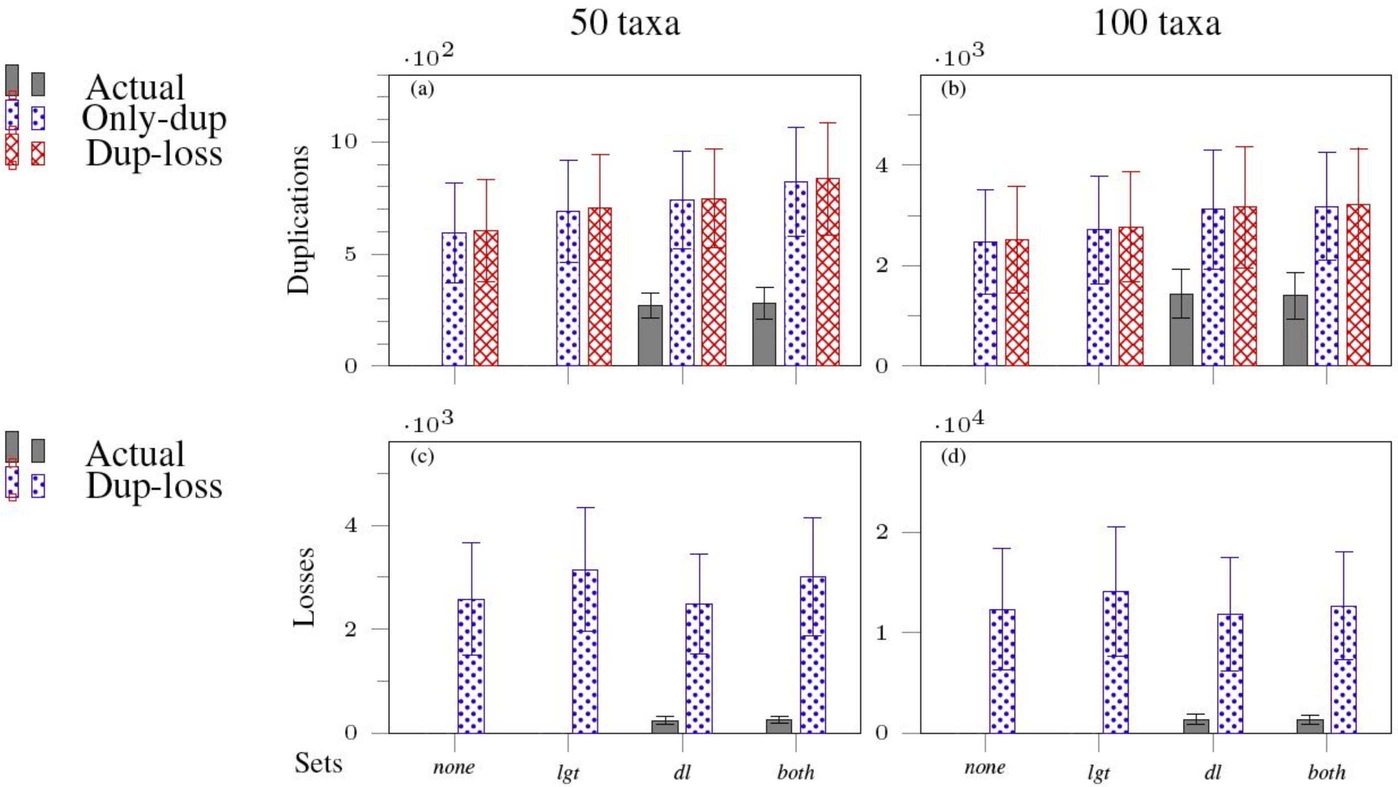


Results - Running Time (Table 1)



Num. Taxa	Sets	Only-dup	Dup-loss	SPRS	MulRF
50	*none*	< 1s	2s	8h 34m 32s	3s
	lgt	< 1s	2s	8h 30m 30s	2s
	dl	< 1s	3s	NA	6s
	both	< 1s	3s	NA	6s
100	*none*	9s	37s	21h 34m 25s	58s
	lgt	11s	49s	19h 6m 9s	51s
	dl	9s	30s	NA	1m 11s
	both	11s	37s	NA	1m 15s

Results - # of Duplication / Loss Events (Figure 8)



Conclusions

MulRF heuristic is

- Faster than SPRS
- Similar speed to GTP methods
- **More accurate** than SPRS and GTP methods
- **Non-parametric** and thus may be more robust to “bad” input data
    - Other sources of gene tree discord, for example, gene tree estimation error
    - Other sources of missing data (other than true loss events), for example, sampling error

Questions about the paper

MulRF implementation:

- How does MulRF compute the initial supertree?
- Does MulRF search from multiple initial supertrees and return the supertree with the best score?
- Does MulRF continue searching from the best tree in the SPR neighborhood until some convergence criterion is reached?

How is the LCA calculated efficiently? Is it just recalculated for every SPR move?

Fin.