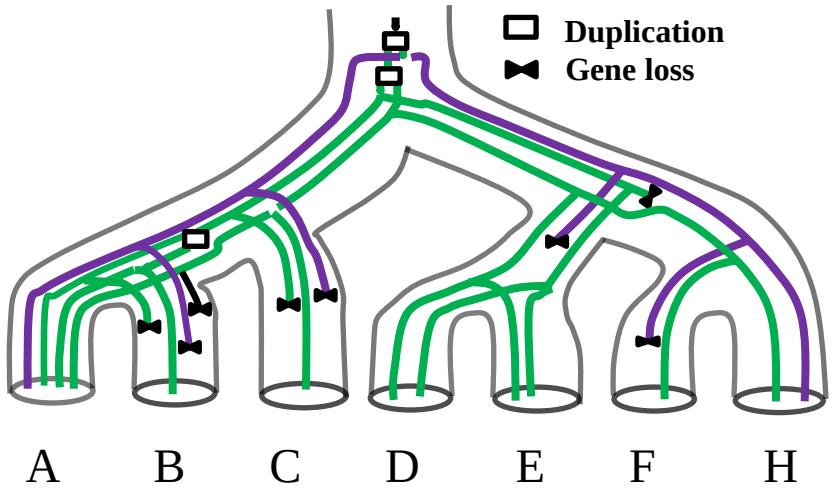
Non-binary Tree Reconciliation

Louxin Zhang
Department of Mathematics
National University of Singapore
matzlx@nus.edu.sg

Introduction: Gene Duplication Inference

Consider a duplication gene family $\mathcal{G}$

Species	Genes
A	A_g1, A_g2, A_g3, A_g4
B	B_g
C	C_g
D	D_g1, D_g2
E	E_g1, E_g2
F	F_g
H	H_g

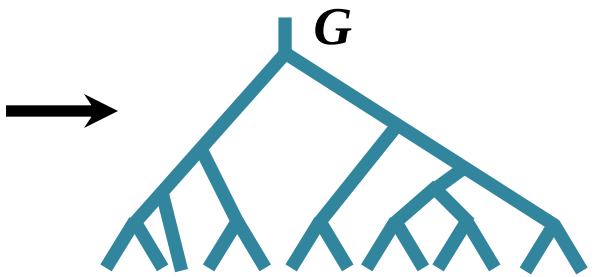


Question: How to reconstruct the duplication history of the gene family $\mathcal{G}$?

Introduction: Tree Reconciliation Approach

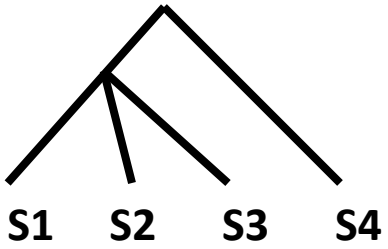
Step 1 Build the gene tree ***G*** for the gene family using gene sequences, and the species tree ***S*** if it is not available.

```
DADSRPKFRELIIEFSKMARDPQRYLVIQGDERMH
DADSRPKFRELILEFSKMARDPQRYLVIQGDERMH
DADSRPKFRELIAEFSKMARDPPRYLVIQGDERMH
DAESRPRFRELIAEFTKMARDPSRYLVIQGDDRMH
DADARPTFKQLAETFAEKARDPGRYLMIPGDKFMR
DAAMRPTFKQLTTVFAEFARDPGRYLAIPGDKFTR
NPESRPSFVLLREKFKQFCSCPDIYVLDRHHPRRM
NPEARPSFTLLKETFQNYCKAPHQYVTEYHLHNKM
DPKSRPGFEILYERFKEFCKVPQLFLENSNKISES
EPEKRPSFEDLVYQFNTMMISPKKYVKIKKTRTLR
```

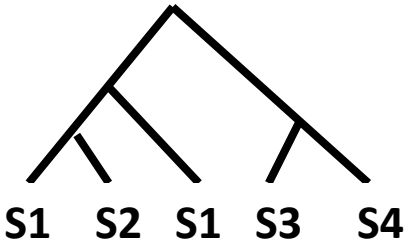


Gene Tree and Species Tree

- A species tree S represents the evolutionary pathways of of a group of species



Species tree *S*



Gene tree *G*

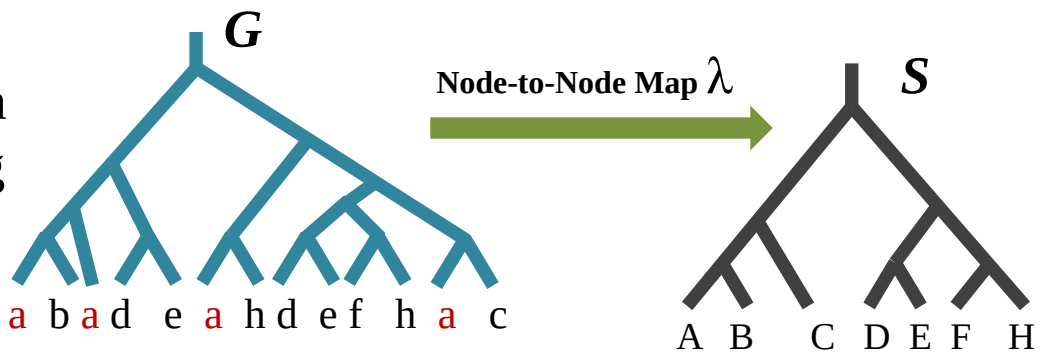
- A gene tree G is reconstructed from gene sequences, representing evolutionary relationship of genes, but is not the duplication history of the gene family.

- G can differ from the corresponding S in two respects.
  - -- The divergence of two genes may predate the divergence of the corresponding species
  - -- Their topologies are different

Introduction: Tree Reconciliation Approach

Step 1 Build the gene tree ***G*** for the gene family using gene sequences, and the species tree ***S*** if it is not available.

Step 2 Reconcile ***G*** and ***S*** to infer gene duplication and loss events, forming a duplication history of the gene family.



LCA reconciliation λ: Binary trees

In G, the leaves are labeled with corresponding species;

$l(x)$: the label of a leaf x of G;

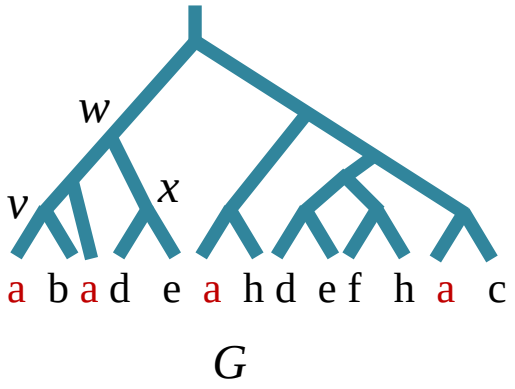
$l^{-}(y)$: the leaf of S that has the label y;

lca: the lowest common ancestor of two nodes

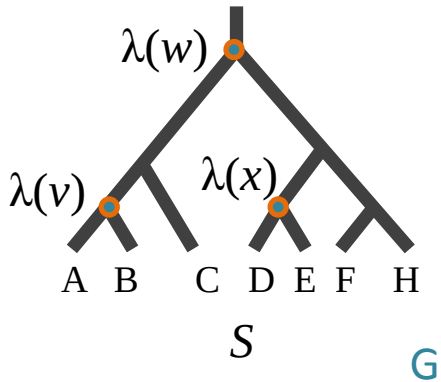
v_1, v_2: the children of v.

λ: $V(G)$ → $V(S)$ is defined as:

$$\lambda(v) = \begin{cases} l_S^{-}(l_G(v)), & v \text{ is a leaf of } G, \\ \text{lca}\{\lambda(v_1), \lambda(v_2)\}, & \text{otherwise} \end{cases}$$



G

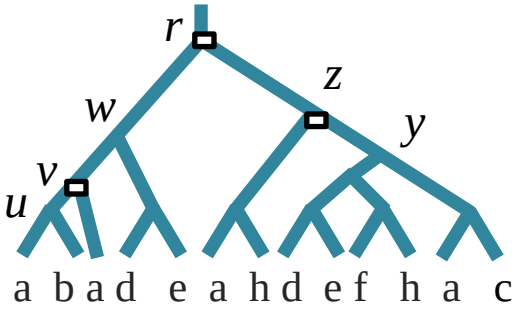


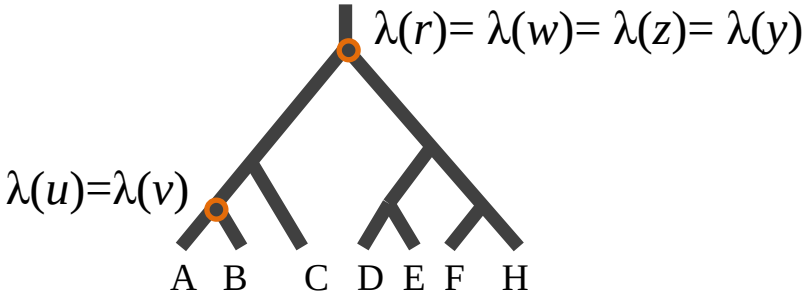
S

Goodman et al, 1979

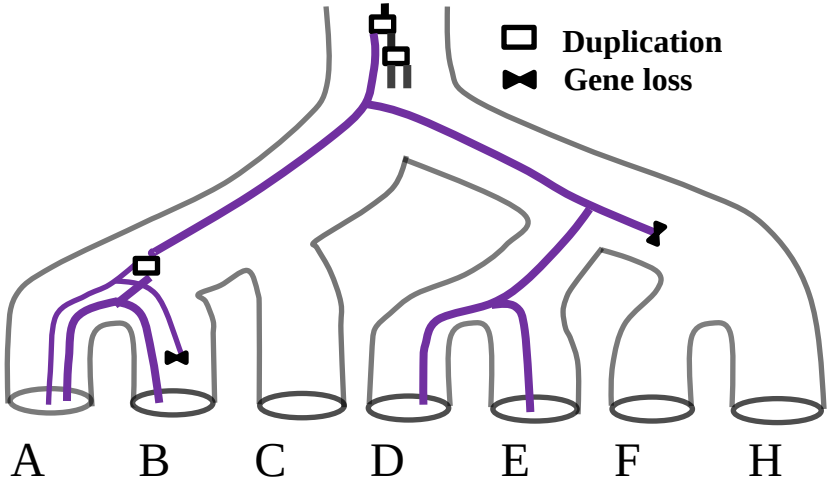
LCA reconciliation λ: Binary trees (con't)

$v \in V(G)$ is a duplication node if $\lambda(v) = \lambda(v_1)$ or $\lambda(v) = \lambda(v_2)$.



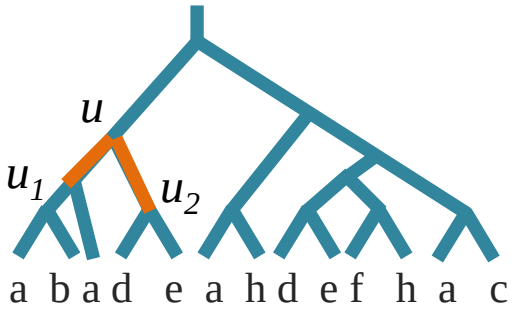


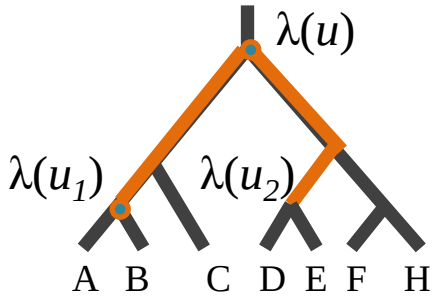
For each duplication node *v*, a duplication is assumed in the branch entering $\lambda(v)$, producing two gene copies, which are the ancestors of the modern genes in the left subtree and in the right subtree, respectively.



LCA reconciliation λ: Binary trees (con't)

(The gene duplication cost of λ) = (no. of duplication nodes)
(The gene loss cost of λ) = (no. of gene loss events)





- The gene loss cost can be computed from the no. of lineages branching off the paths from λ(u) to λ(u_1) and λ(u_2)

- Both gene duplication and loss costs are two dissimilarity measures for gene and species trees.

Theorem Let G and S be binary.

1). λ gives a duplication history of the gene family with the least gene duplication events (Gorecki & Tiuryn, 2006).

2). λ gives a duplication history of the gene family with the least gene loss events (Chauve & El-Mabrouk, 2009).

3). λ gives a duplication history of the gene family with the least deep coalescence cost (Wu & Zhang, 2011).

4). λ is linear-time computable (Zhang, 1997, Chen, Durand & Farach 2000).

λ is the parsimonious reconciliation for binary trees

Introduction: Species Tree Reconstruction

Species Tree (ST) Problem

Instance: A set of gene trees G_i $(0 \leq i \leq n)$ and a cost function c().

Solution: A binary species tree S that minimizes $\sum_{1 \leq i \leq n} c(G_i, S)$

The following cost functions have been used:

-- Gene duplication cost W
-- Gene loss cost L
-- Deep coalescence cost DC
-- Mutation cost (W+L), or weighted sum of W and L
-- Robinson-Foulds distance

- The ST problem is NP-hard for each of the above cost functions.

Hallett & Lagergren, 2001
Yu, Warnow & Nakhleh, 2011
Than & Nakhleh, 2009
Liu, Yu, Kubatko, Pearl & Edwards, 2009

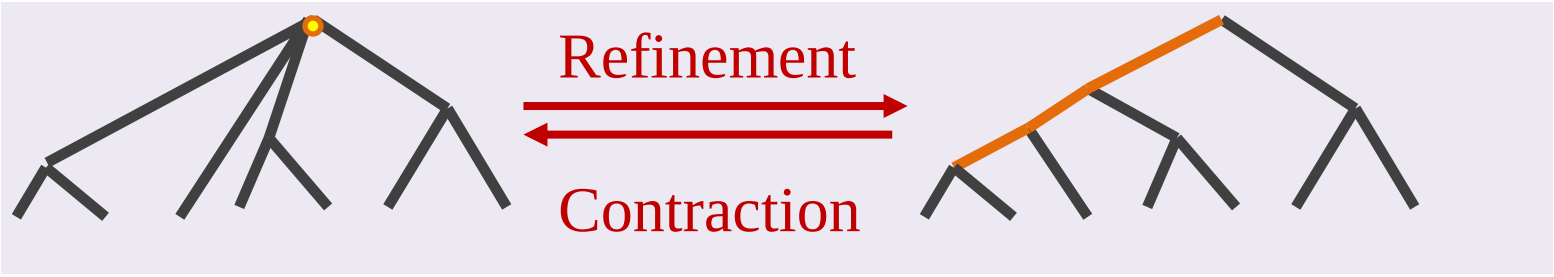
McMorris & Steel, 1993
Ma, Li, & Zhang, 2000;
Bansal & Shamir, 2010;
Zhang, 2011;

Introduction: Unify Two Problems

General Reconciliation (GR) Problem

Instance: A gene tree G and a species tree S and a reconciliation cost c(,).

Solution: A binary refinement $\hat{G}$ of G and $\hat{S}$ of S such that the lca reconciliation of $\hat{G}$ and $\hat{S}$ minimizes a reconciliation cost c($\hat{G}$, $\hat{S}$).



Eulenstein, Huzurbazar, Liberles, 2010

Two remarks

1. The GR problem is a generalization of binary tree reconciliation
2. The species tree inference problem is a special case of the GR problem, and hence the latter is NP-hard.
    - Set S be the star tree over the species in the reduction from the Species Tree problem to the GR problem

Species Tree Inference
Instance: A set of gene trees G_i $(0 \leq i \leq n)$.
Solution: A binary species tree S that minimizes $\sum_{1 \leq i \leq n} c(G_i, S)$

Outline of Today's Talk

- Relationship between tree similarity measures
- Algorithms for the General Reconciliation problem
  - -- Extensions of the reconciliation of binary trees to non-binary gene trees
  - -- Exact algorithm for reconciling two non-binary trees
- Computer program TxT
- Conclusion

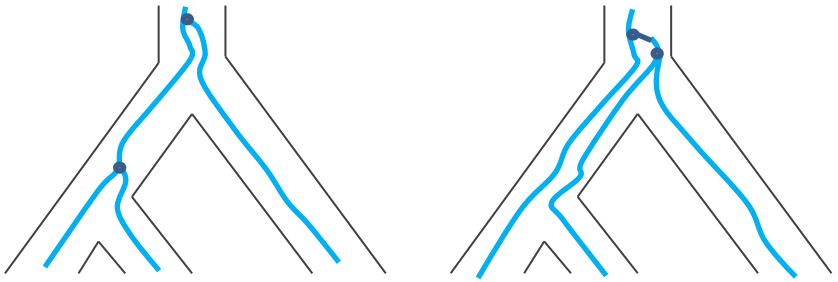
Zheng, Wu & Zhang, 2011
Zheng & Zhang, 2013

Part I: Relationship between Cost Functions

Theorem Let S be a species tree and G the gene tree of a gene family. If one family member is found in each of the species, then

$$C_{\text{loss}}(G,S) = 2C_{dup}(G,S) + C_{dc}(G,S)$$

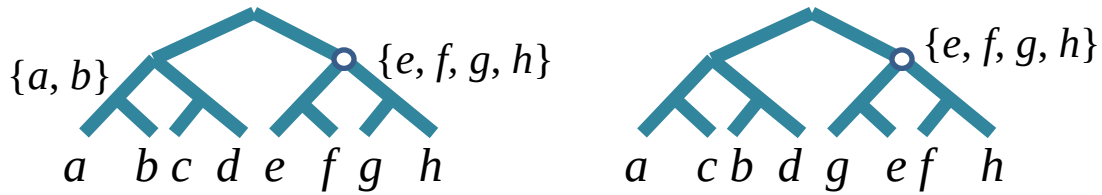
where $C_{dc}(G,S)$ (deep coalescence cost) is defined as the sum of extra lineages in all branches when G is mapped onto S.



Maddison, 1997
Zhang, 2011

Consider two singly-labeled trees G and S over n taxa X (that is, each leaf is uniquely labeled with $e \in X$).

The Robinson-Foulds distance $C_{\mathrm{RF}}(G,S)$ is defined to be the number of leaf clusters appearing in G but not in S.



Proposition (i) For G and S defined above,

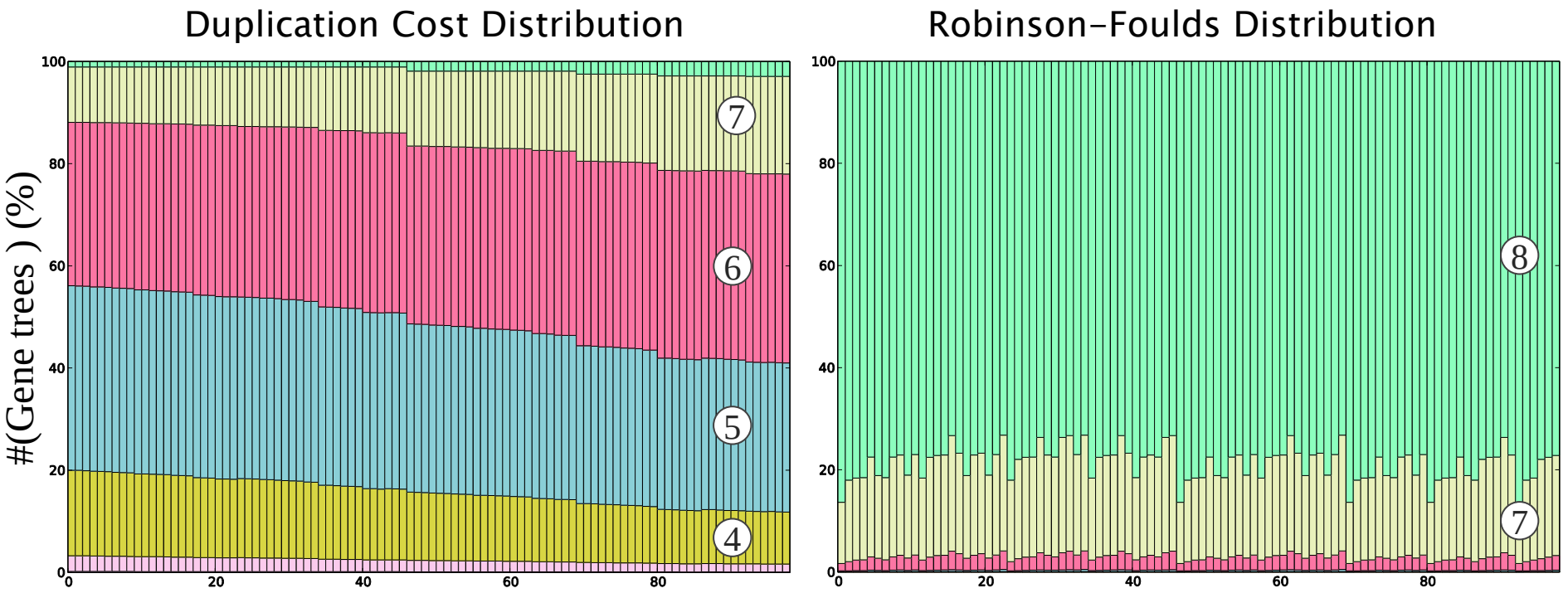
$$C_{\mathrm{dup}}(G,S) \le C_{\mathrm{RF}}(G,S) \le C_{DC}(G,S) \le C_{\mathrm{loss}}(G,S).$$

(ii) $\max_{G,S} C_{\mathrm{dup}}(G,S) = \max_{G,S} C_{\mathrm{RF}}(G,S) = n-2.$

Theorem (i) There exist G and S with n leaves such that $C_{\mathrm{dup}}(G,S)=1$, but $C_{\mathrm{RF}}(G,S)=n\text{-}2$.

(ii) For any G and S defined above,

$$max\left(C_{dup}(G,S), C_{dup}(S,G)\right) \geq \sqrt{C_{\mathrm{RF}}(G,S)}.$$



98 species tree topologies for 10 taxa (listed in Fumas rank)

Part II: Reconciling Non-binary G and Binary S

Instance: A gene tree G and a binary species tree S and a cost c( ).
Solution: The binary refinement $\hat{G}$ of G such that the lca reconciliation of $\hat{G}$ and S minimizes c($\hat{G}$, $\hat{S}$).

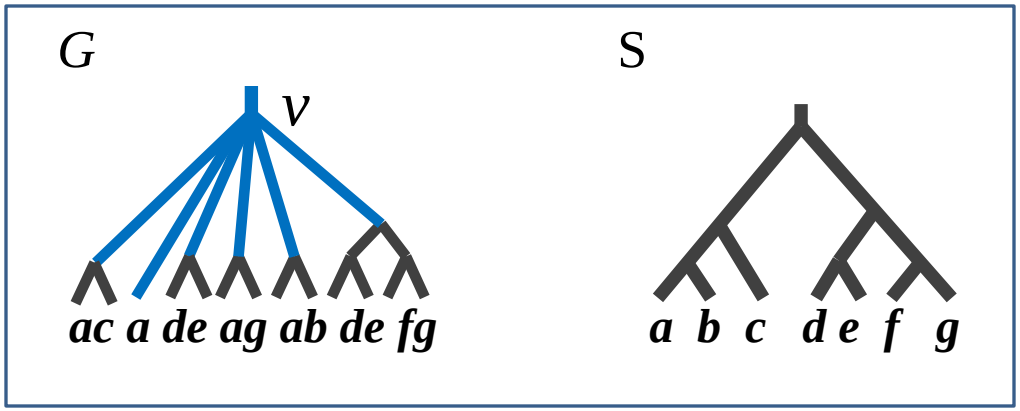
- The following duplication inference rule does not work for non-binary nodes:

> A duplication is associated with u having children u_1 and u_2 iff $\lambda(u) = \lambda(u_1)$, or $\lambda(u) = \lambda(u_2)$.

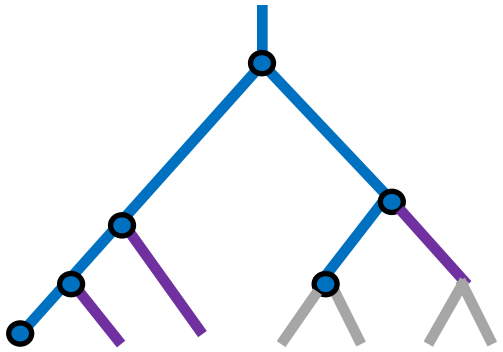
- Durand et al (2006) presented first dynamic programming alg. for reconciling a non-binary gene tree and a binary species tree.

- Generalize the reconciliation to non-binary gene trees. The whole process takes O(|G|+|$\hat{S}$|) time for the duplication and loss costs.

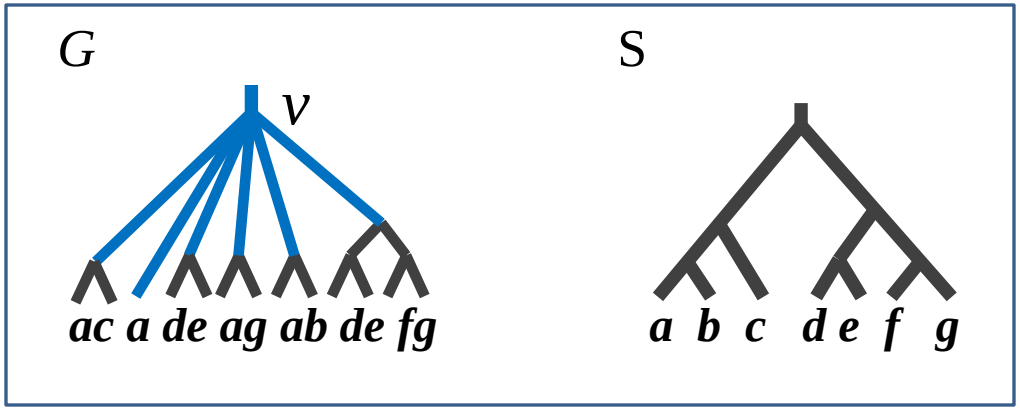
λ: The lca reconciliation of *G* and *S*

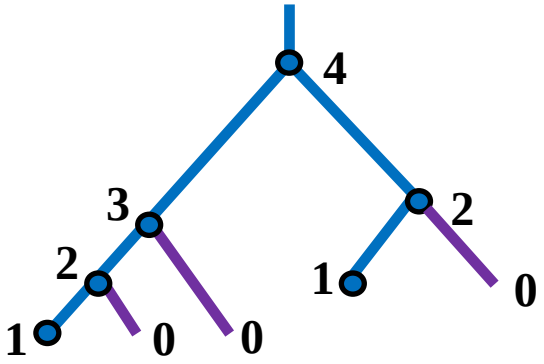


- The node *v* and its children are mapped to a subtree (blue) under λ, which is expanded into a binary subtree (by adding purple edges).



The image subtree I(*v*)
($I^+(v)$ after extension)





Step 1 Compute ***m*(*u*)**, the maximum number of child images in a path from u to some leaf descendant in $I^{+}(v)$.

Algorithm

ω(u) is the # of children mapped to u.

$$m(u) = max\{m(u_1), m(u_2)\} + \omega(u).$$

Thm (i) The min. dup. cost for refining the non-binary node v is $m(\lambda(v)) - 1$.

(ii) The min. loss cost for refining v is equal to (# of purple edges).

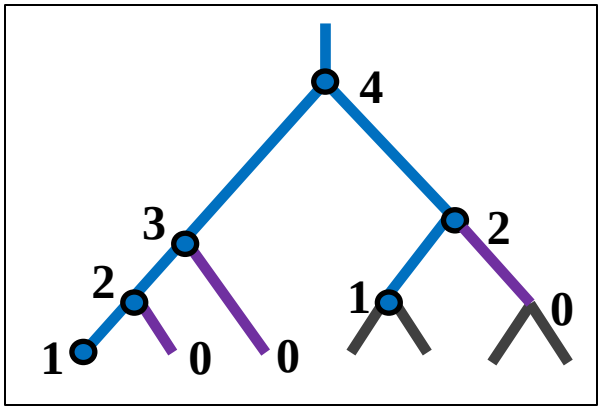
Idea of Proof.

$$\mathcal{P} = (\{\lambda(v_1), \lambda(v_2), \ldots, \lambda(v_k)\}, \subseteq)$$

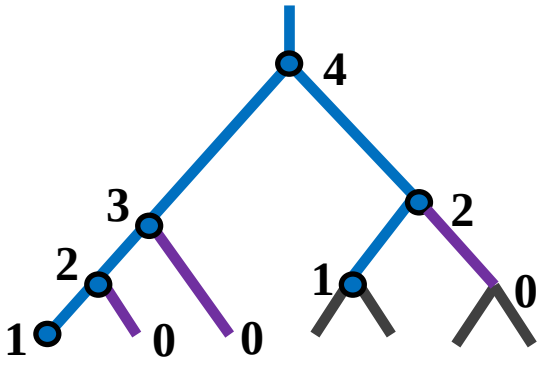
- L: The size of the longest chain in $\mathcal{P}$, which is $m(\lambda(v))$ in our case ;
- P: The min. # of antichains into which $\mathcal{P}$ may be partitioned.

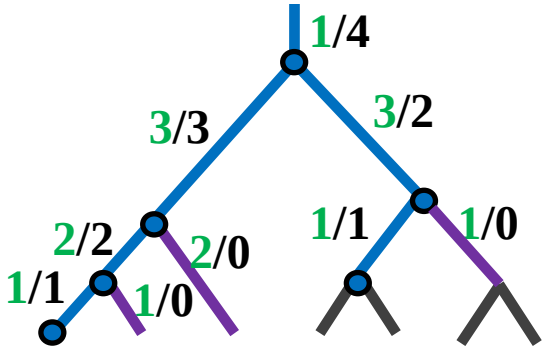
Dual of Dilworth Theorem (Mirsky, 1971): L=P.

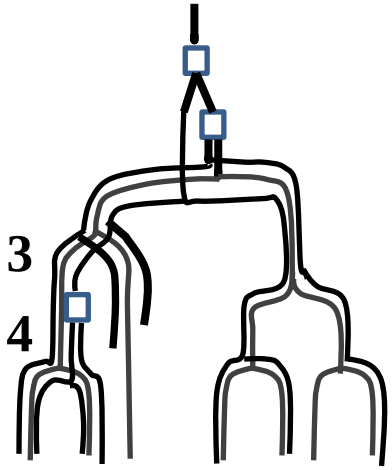
(ii) It is obvious.



1. A Simple Refinement with the Optimal Dup. Cost







Step 2 Compute ***α(u)*** / ***β(u)*** using $m(u)$.

Algorithm

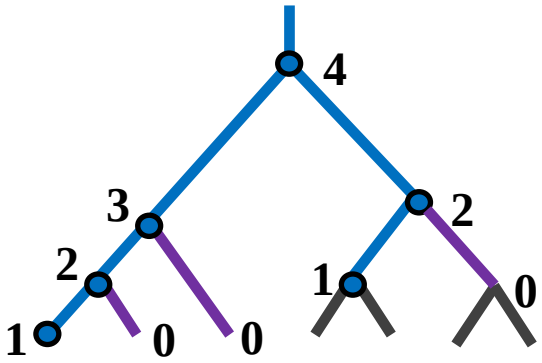
$\alpha(r) = 1, \quad \beta(r) = \mathrm{m}(r);$

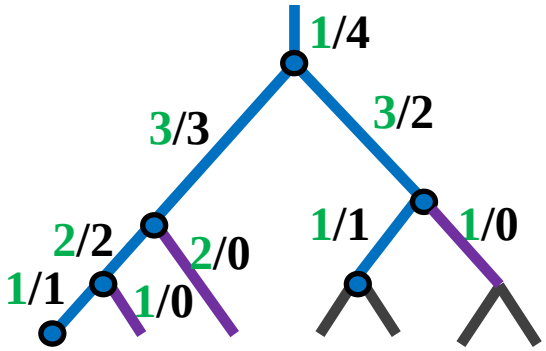
$\alpha(u) = \beta\big(p(u)\big) - \omega\big(p(u)\big),$
$\beta(u) = m(u).$

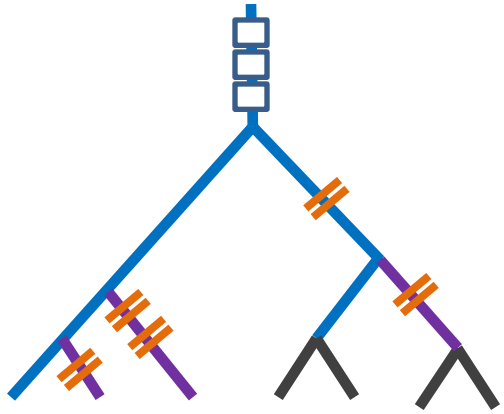
α(u): the # of genes flowing into a branch $(p(u), u)$.

***β(u)*:** the # of genes leaving a branch $(p(u), u)$.

ω(u): the # of children mapped to u.





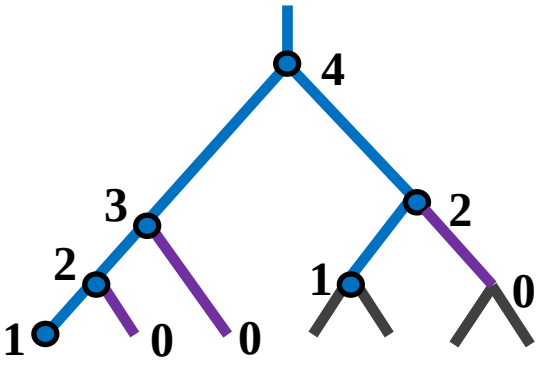


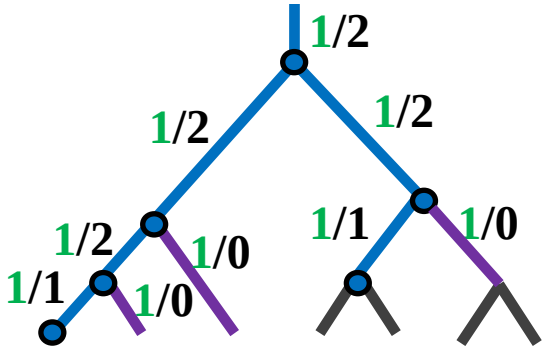
Step 3 Infer duplications and losses:

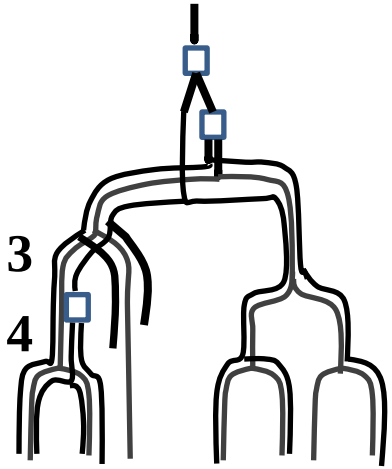
If $\alpha(u) < \beta(u)$, duplications (▭) are postulated.

If $\alpha(u) > \beta(u)$, losses (〓) are postulated.

2. A Simple Refinement with the Optimal Loss Cost







Step 2 Compute ***α(u)*** / ***β(u)*** using $m(u)$.

Algorithm

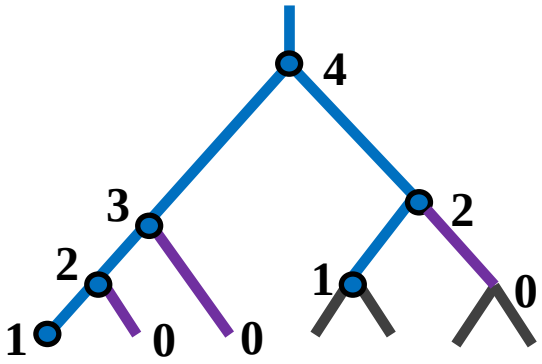
$$\alpha(u) = 1,$$

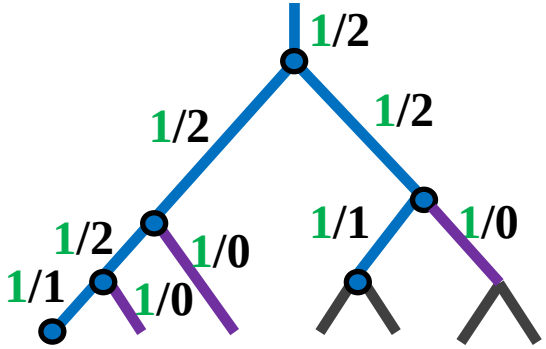
$$\beta(u) = \begin{cases} \omega(u) + 1, & \text{if } u \text{ is an internal node} \\ \omega(u), & \text{if } u \text{ is a leaf} \end{cases}$$

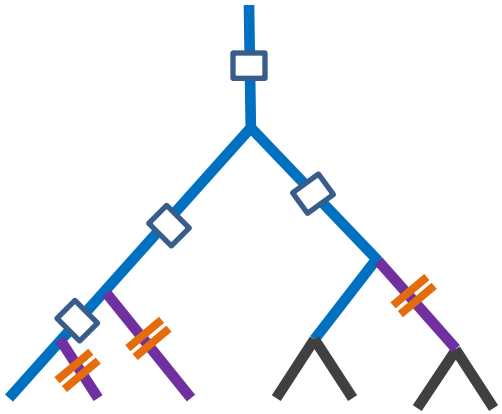
α(u): the # of genes flowing into a branch $(p(u), u)$.

***β(u)*:** the # of genes leaving a branch $(p(u), u)$.

ω(u): the # of children mapped to u.





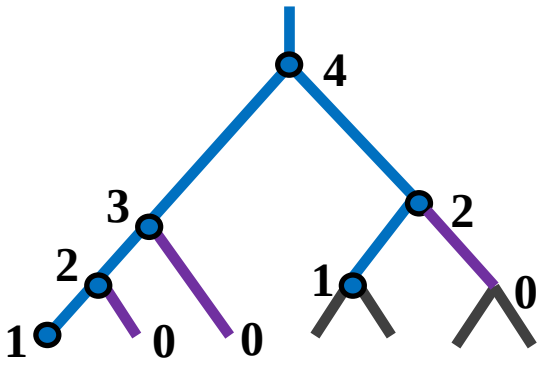


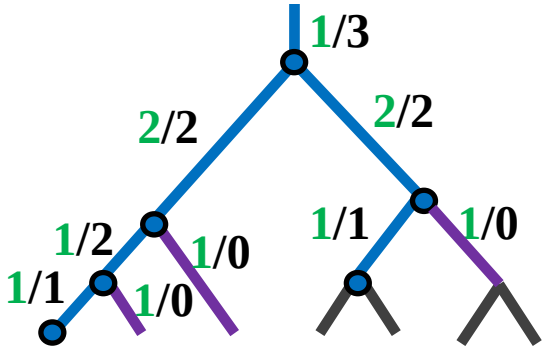
Step 3 Infer duplications and losses:

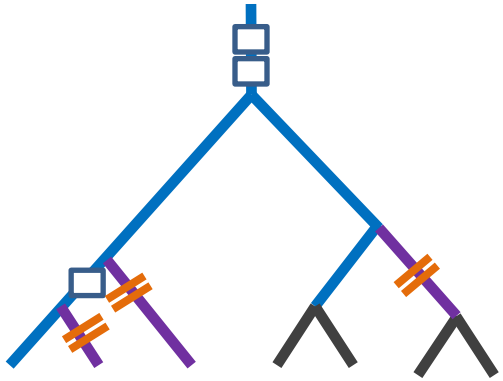
If $\alpha(u) < \beta(u)$, duplications (▭) are postulated.

If $\alpha(u) > \beta(u)$, losses (〓) are postulated.

3. A Refinement Minimizing the Loss Cost with the Constraint of Optimal Dup. Cost







Step2: Compute ***α(u)*** / ***β(u)*** using $m(u)$.

Step 3: Infer duplications and losses.

If $\alpha(u) < \beta(u)$, duplications (▭) are postulated.

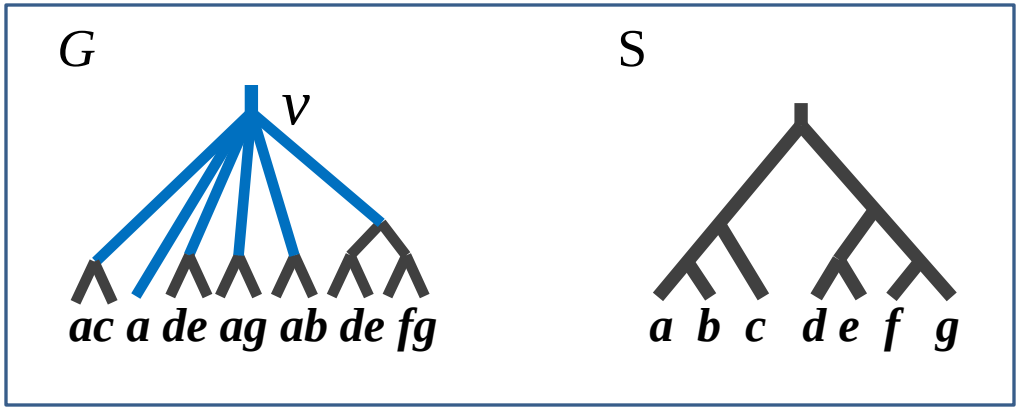
If $\alpha(u) > \beta(u)$, losses (〓) are postulated.

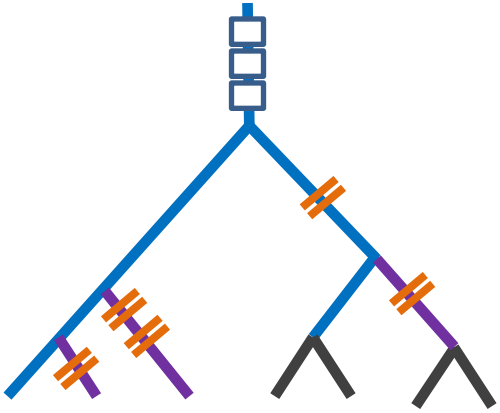
Algorithm

$$\alpha(r) = 1$$

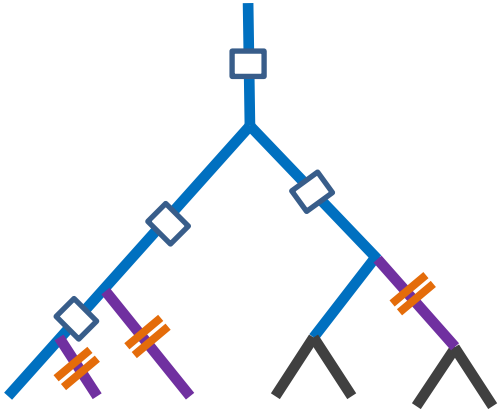
$$\alpha(u) = \beta(p(u)) - \omega(p(u))$$

$$\beta(u) = \begin{cases} m(u) \text{ if } \alpha(u) \geq m(u) \text{ or } u \text{ is a leaf;} \\ \max\{\alpha(u), \min\{m(u_1), m(u_2)\} + \omega(u), 1 + \omega(u)\}. \end{cases}$$

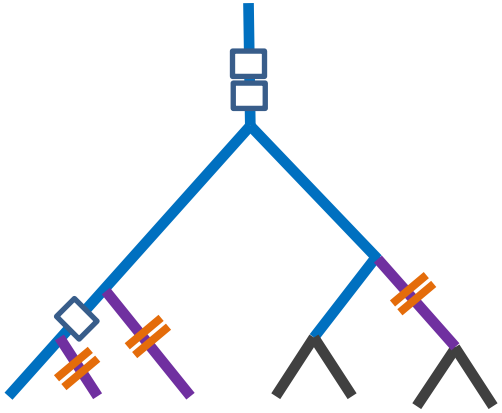




Dup-optimal solution

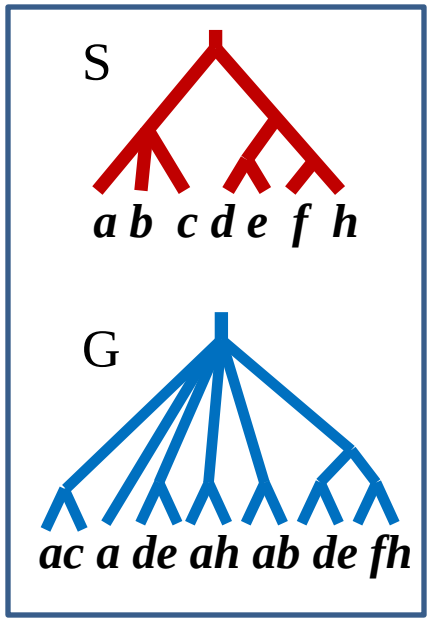


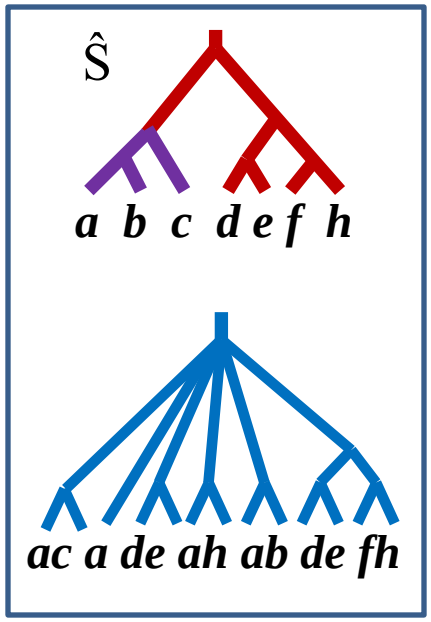
Loss-optimal solution

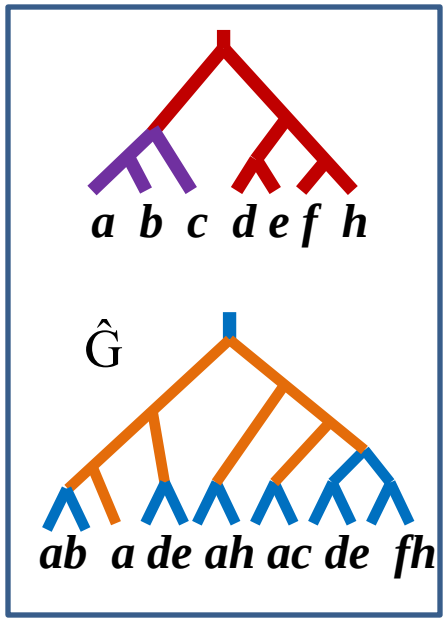


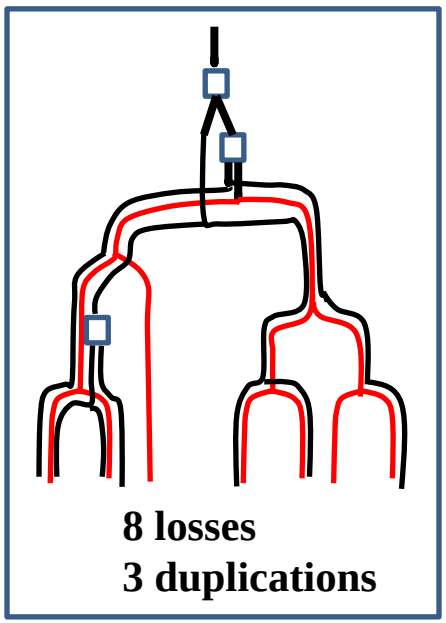
Solution of minimizing duplications and then loss

4. Exact Algorithm for Reconciling Non-binary Trees







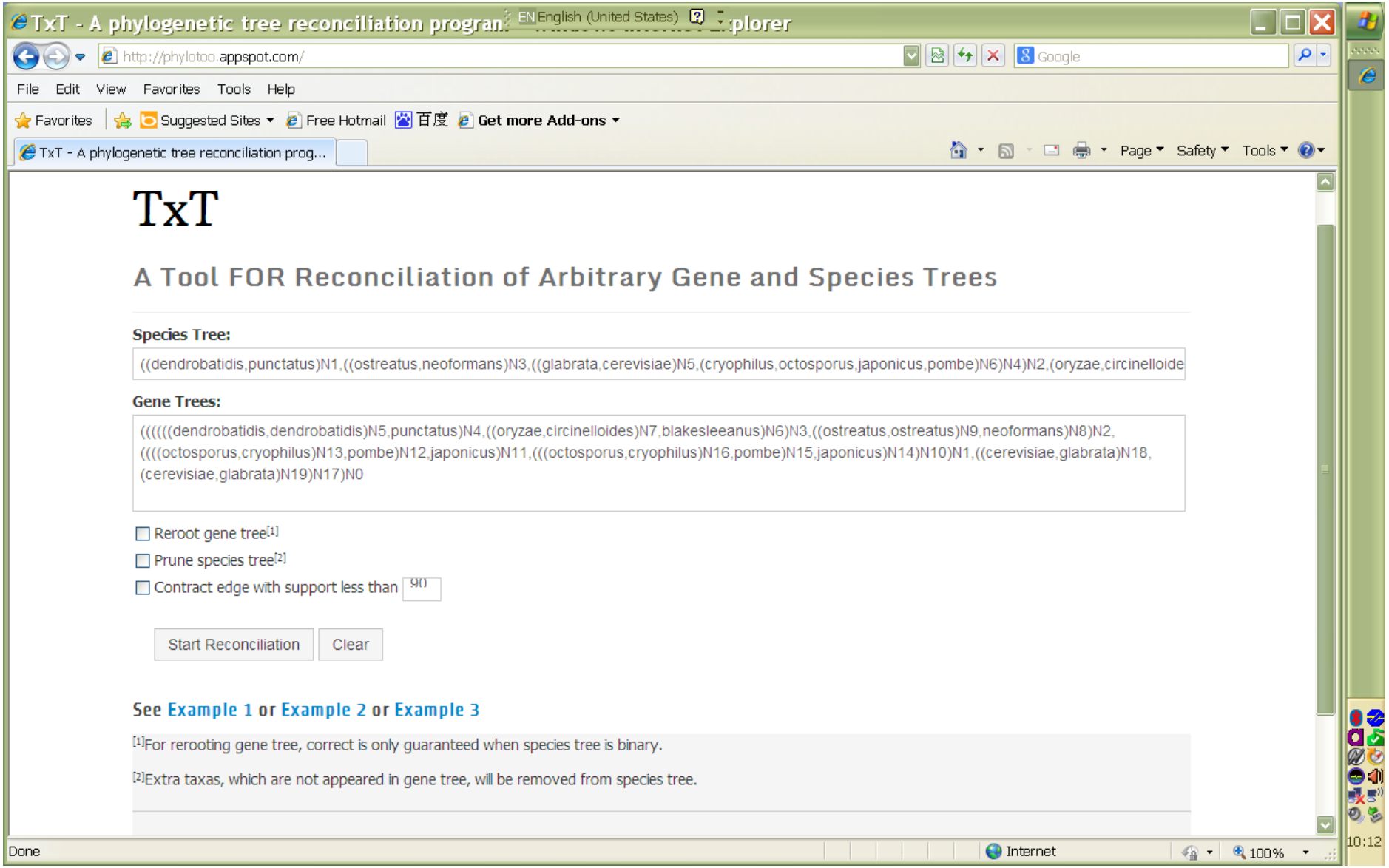


Step 1
Obtain the optimal refinement $\hat{S}$ of S using the union network

Step 2
Refine G based on the refinement $\hat{S}$ of S, obtaining $\hat{G}$

Step 3
Reconcile $\hat{G}$ and $\hat{S}$ to infer the evolution of the gene family

http:phylotoo.appspot.com



Conclusion

- Modeling gene duplication, losses, horizontal gene transfer, incomplete lineage sorting simultaneously

  -- Hallett, Lagergren & Tofigh, 2004

  -- Stolzer et al, 2012

  -- Bansal, EJ Alm, M Kellis, 2012

- Likelihood methods for tree reconciliation

  -- Arvestad, Lagergren, Sennblad, 2009

  -- Boussau et al. 2013

  -- Liu, Yu, Kubatko, Pearl, Edwards, 2009