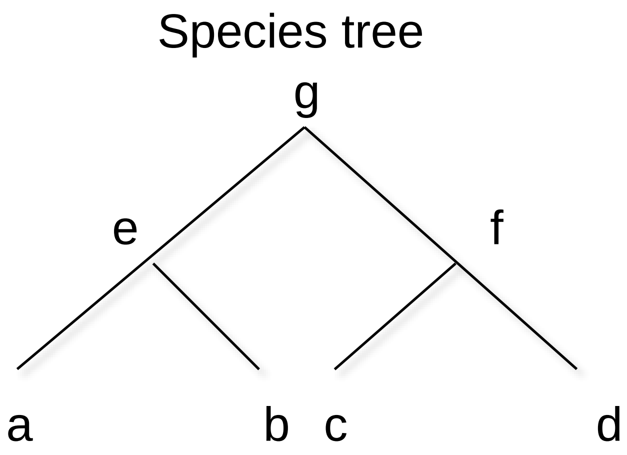
# An optimal reconciliation algorithm for gene trees with polytomies

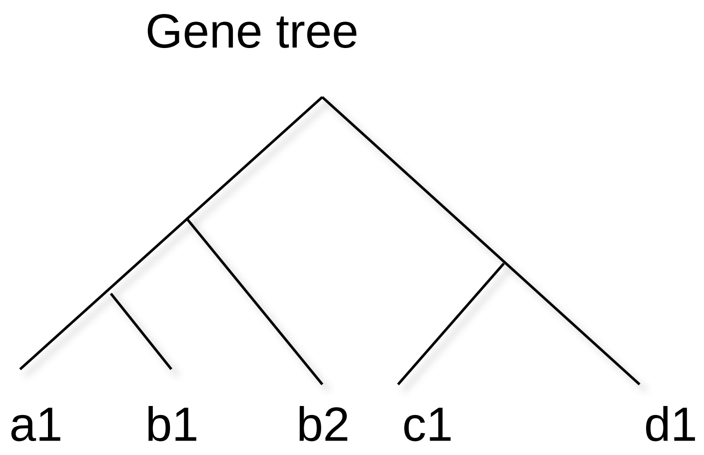
**Manuel Lafond,** Krister M. Swenson, Nadia El Mabrouk
DIRO, Université de Montréal

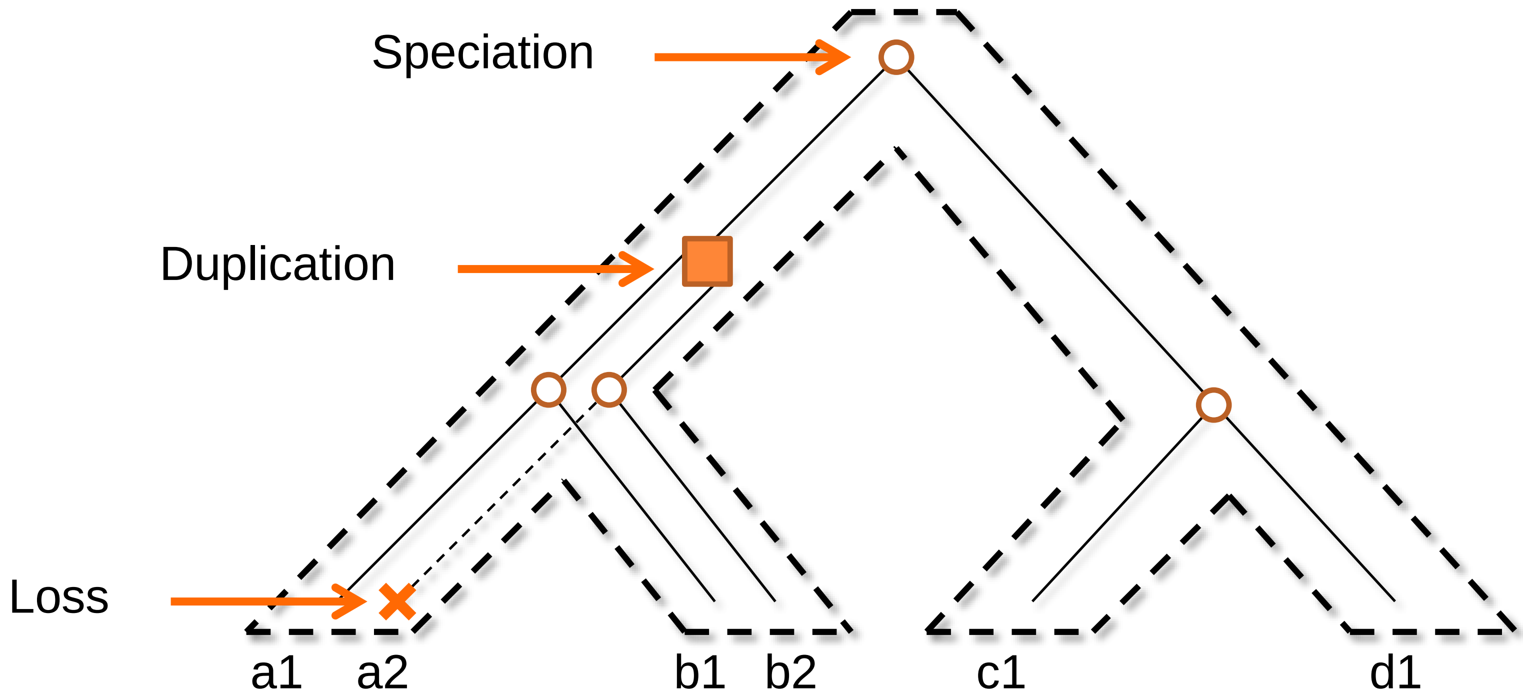
# Introduction

- Gene family
  - Several similar genes that have evolved from a common ancestor
  - Usually identified by sequence similarity
- **Dup-loss model** : Evolution scenario determined by three kinds of events
  - **Speciation** : a new species is created, one copy of the gene existing in both species
  - **Duplication** : the gene is duplicated, giving the species at least two copies of it
  - **Loss** : the gene disappears from the family

# Gene family history

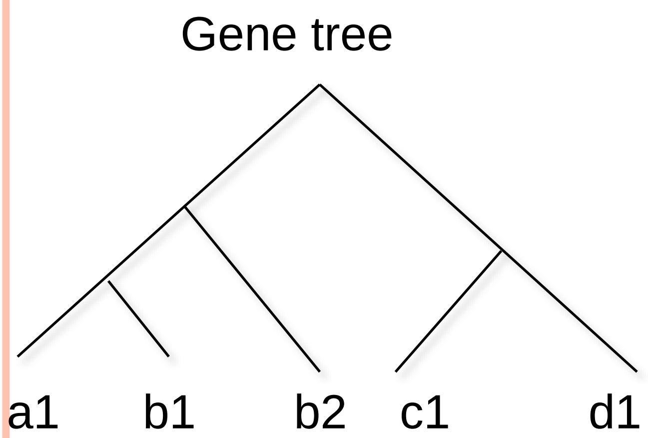


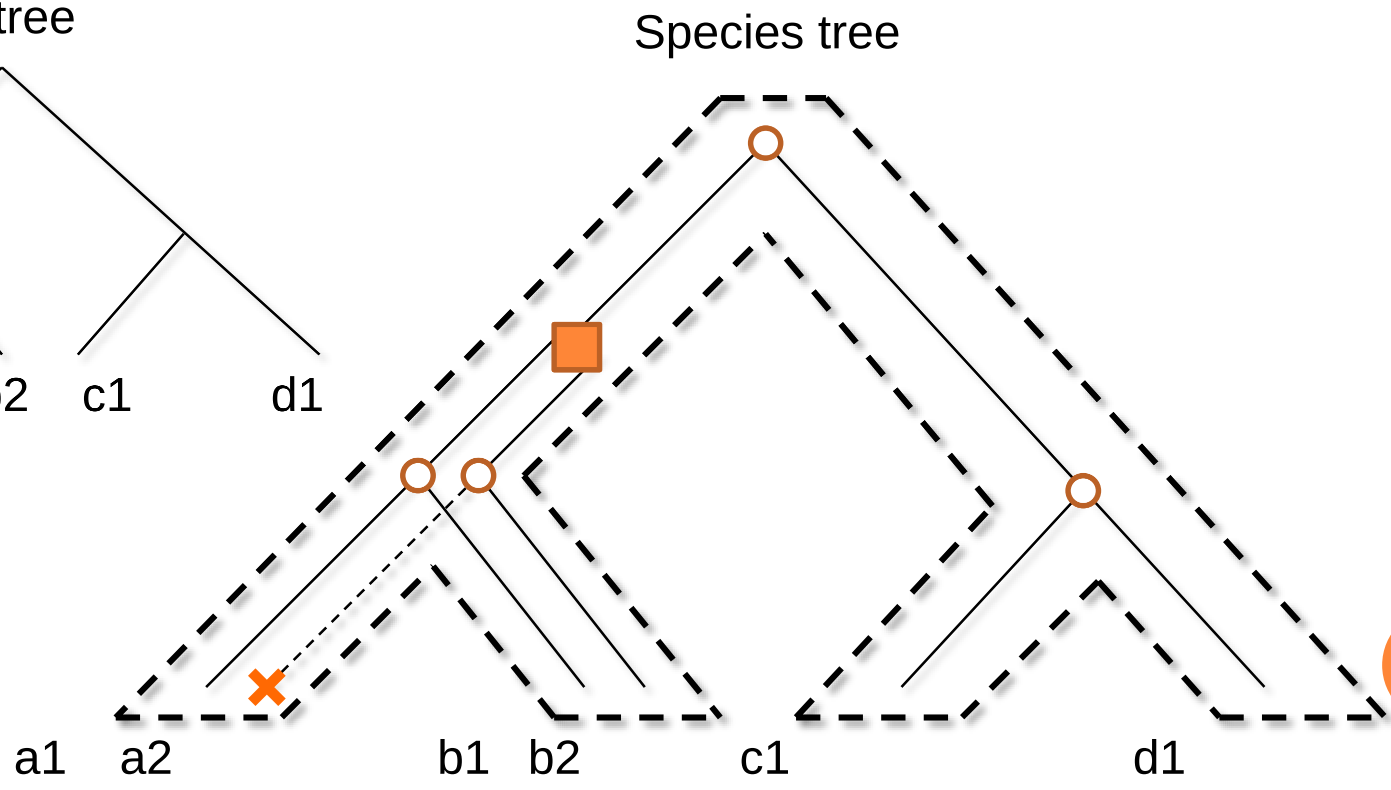



# Reconciliation

- **Given** : a set of genes in the same family, a gene tree G and a species tree S
- **Infer** : the evolutionary events that have led to the observed gene tree

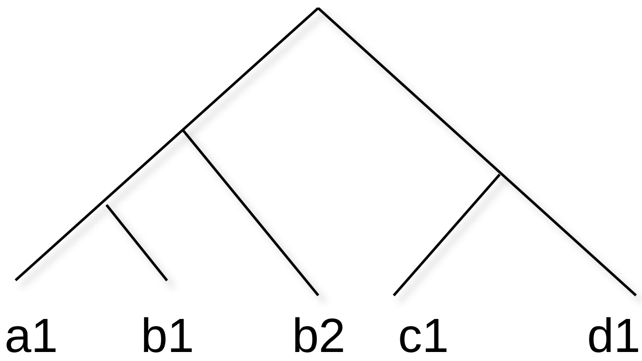


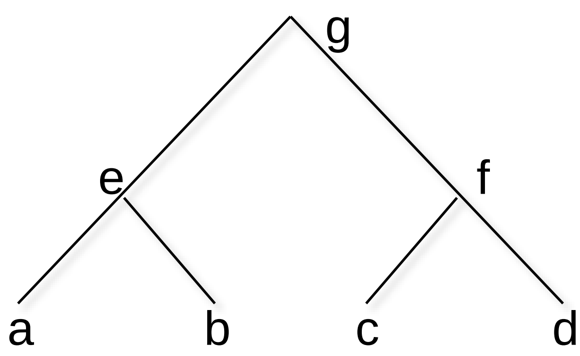
# Reconciliation

- A reconciliation is an « extension » of G that is **consistent** with S i.e. reflects the same phylogeny

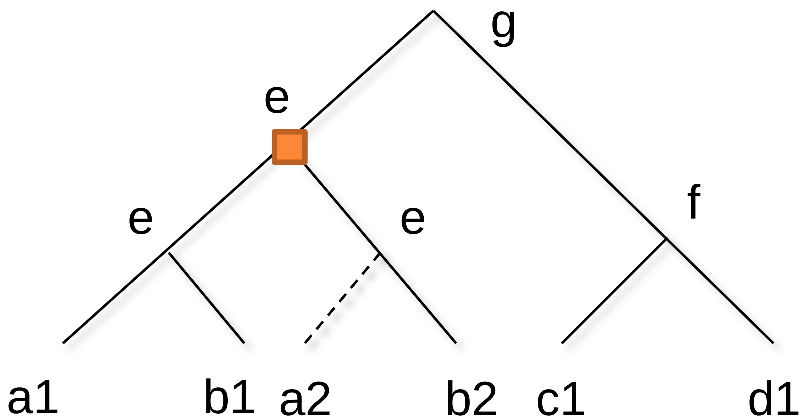
Gene tree




Species tree



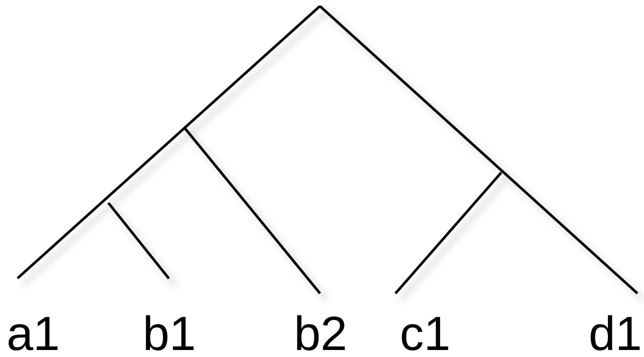

Reconciliation tree



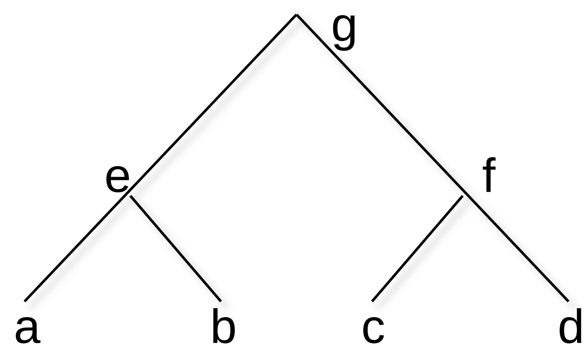
# Reconciliation

- Parsimony criterion : minimum number of duplications + losses (mutation cost)

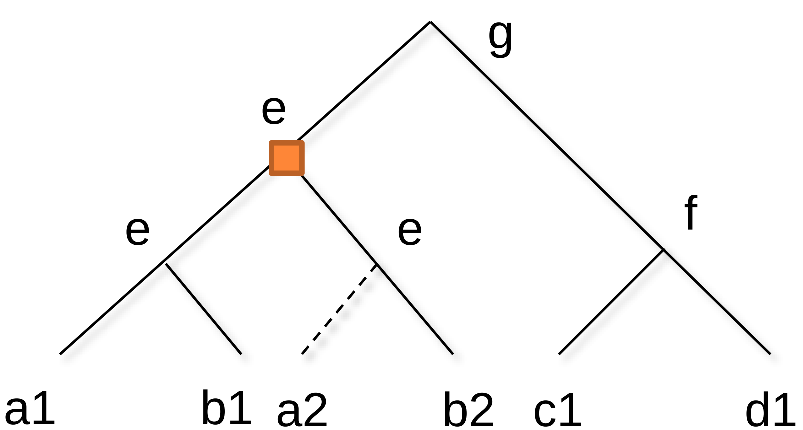
Gene tree




Species tree

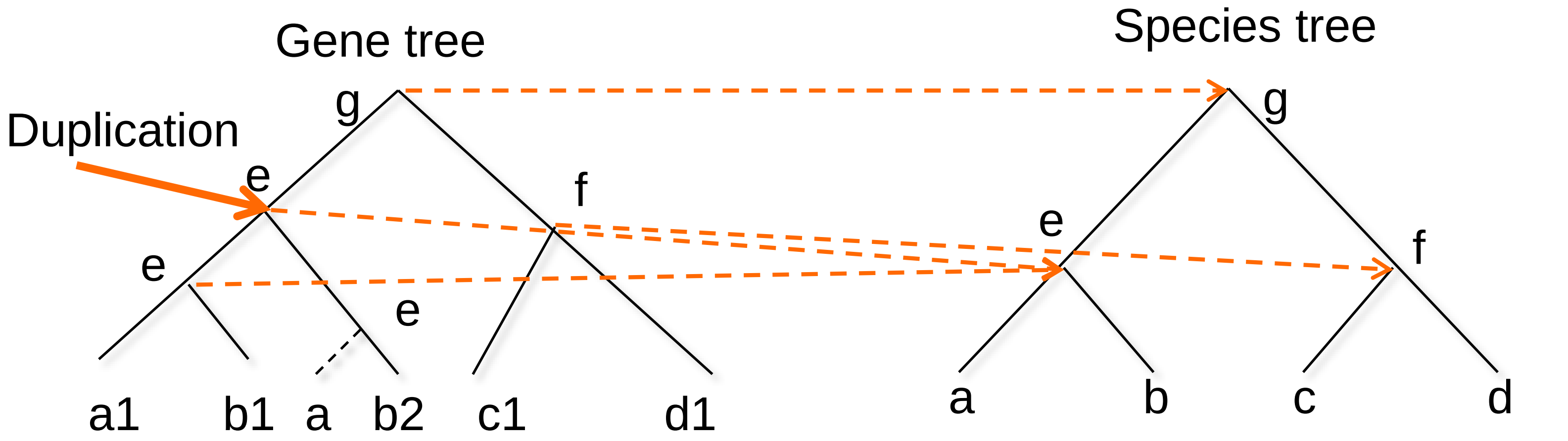



Reconciliation tree



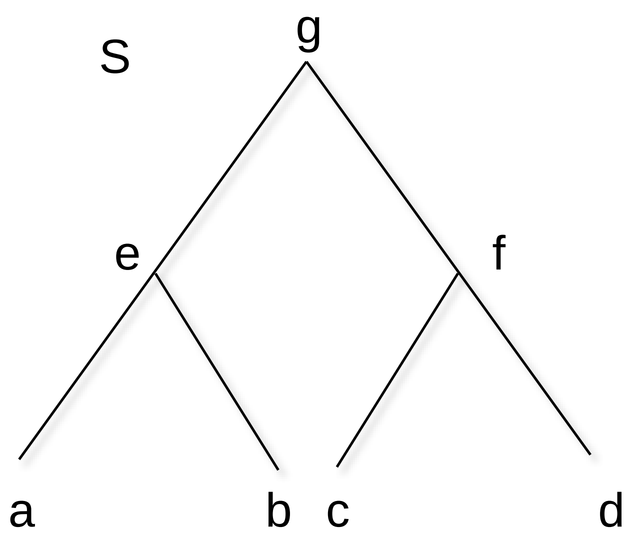
# LCA Mapping

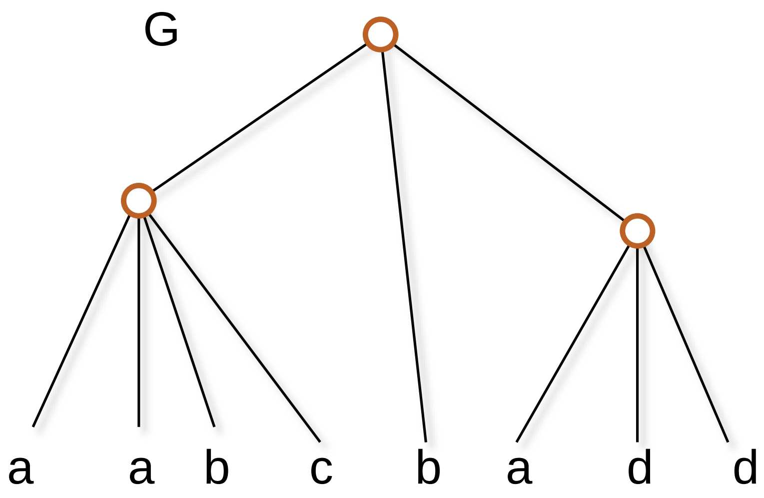
- Many possible reconciliation trees
- LCA Mapping *(Bonizzoni et al., 2003)*
  - Map each node of G with the ***lowest common ancestor*** of its leaves
  - Minimizes the duplication+loss cost in linear time
  - The ***label*** of a node x is the LCA mapping of x



# Motivation

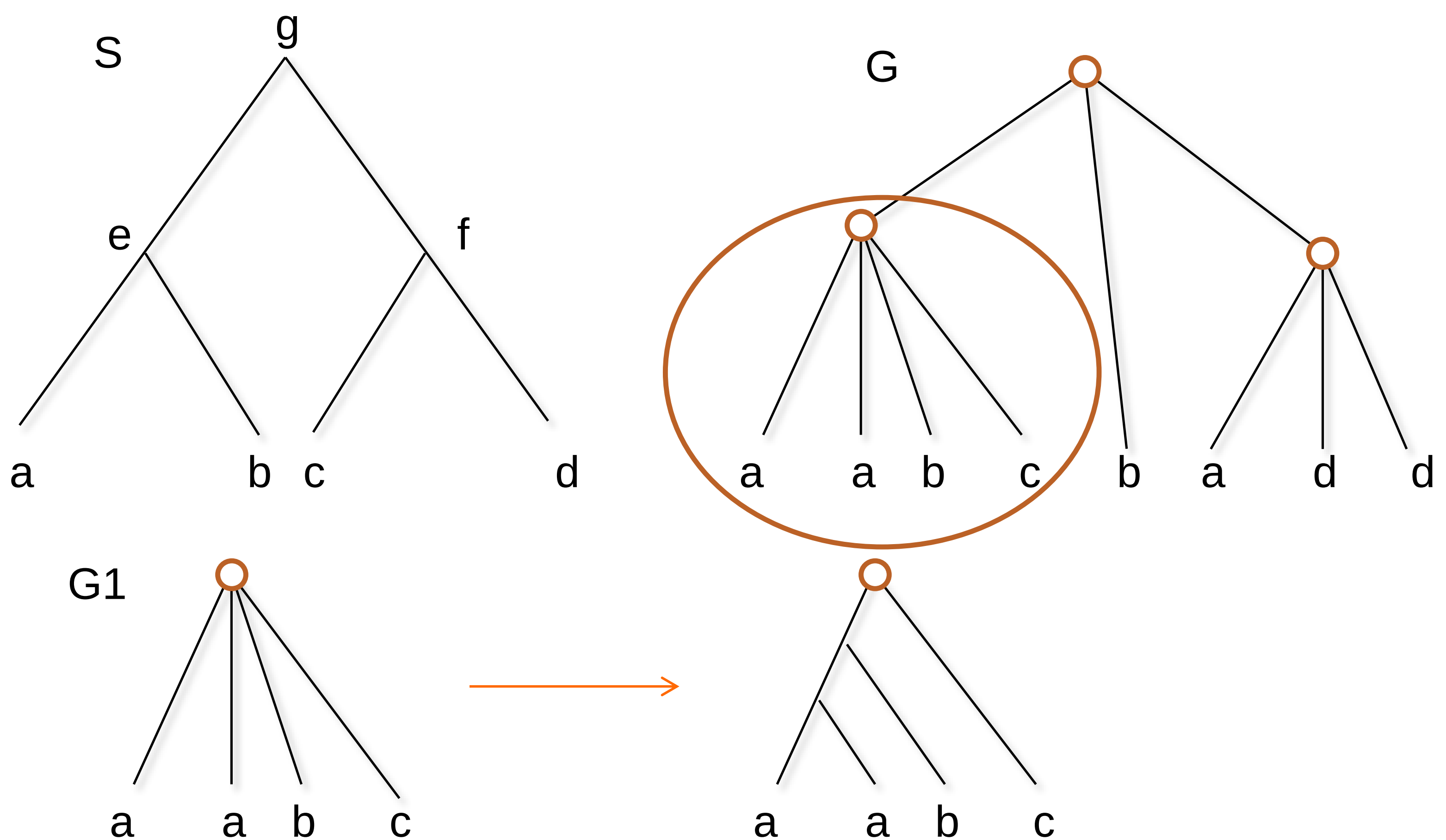
- Most known methods work with binary gene trees
- In case of uncertainty, a gene tree can be non-binary (weak edges)
- Non-binary nodes are called **polytomies**
- Reconciliation trees are binary



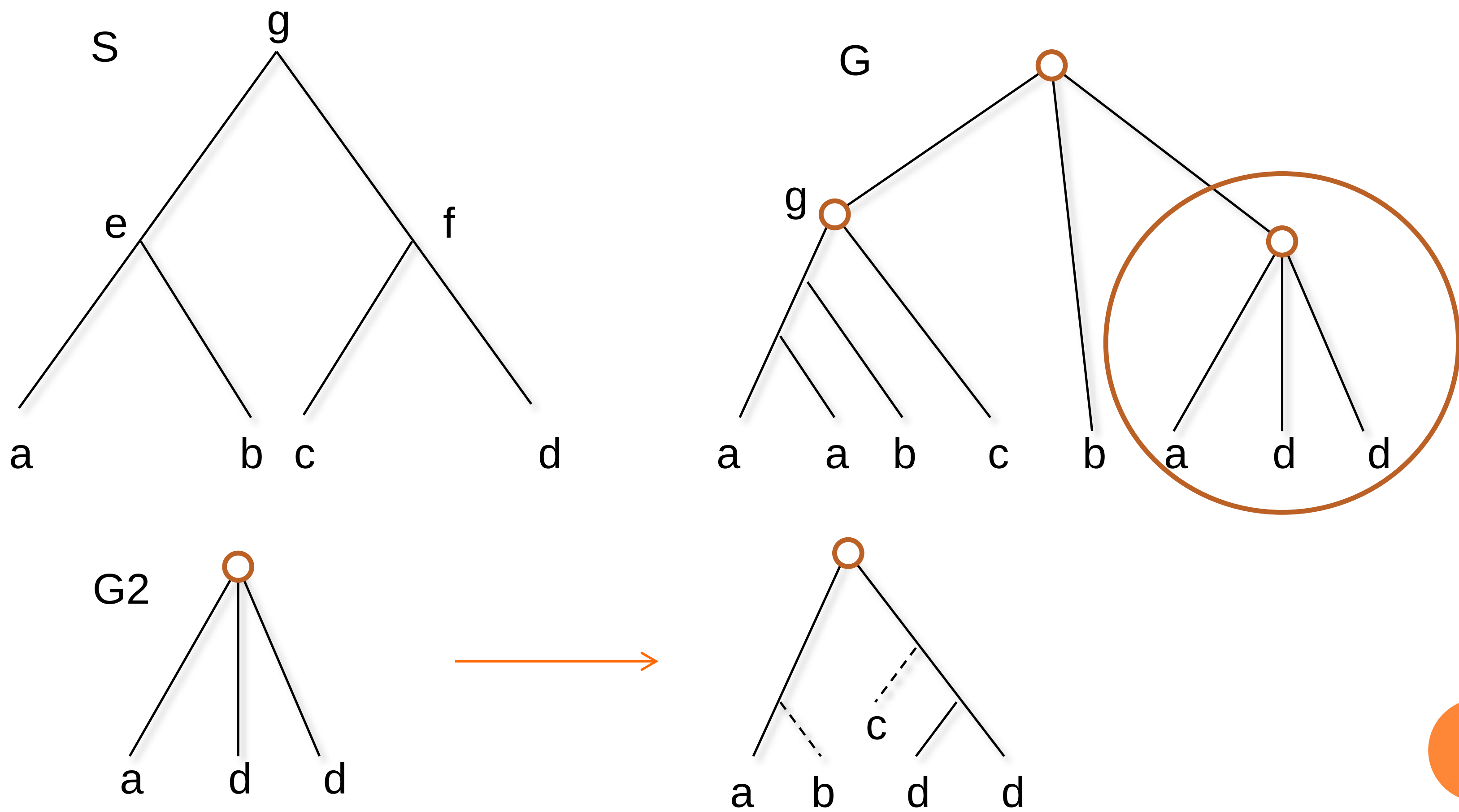
# Polytomies

- Each polytomy can be solved independently *(Chang & Eulenstein, 2006)*
  - Cubic time algorithm for each polytomy



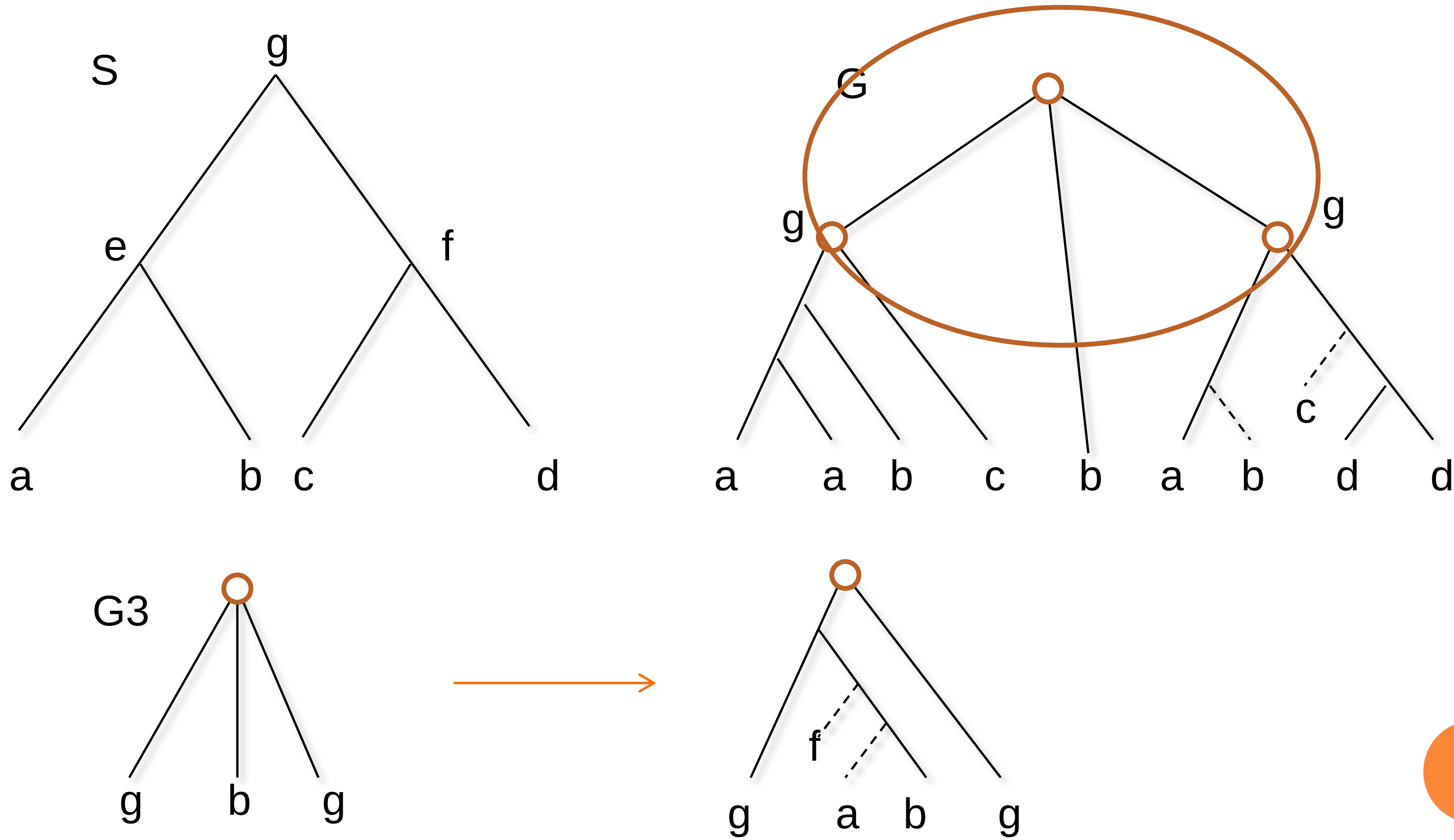
# Polytomies

- Each polytomy can be solved independently *(Chang & Eulenstein, 2006)*



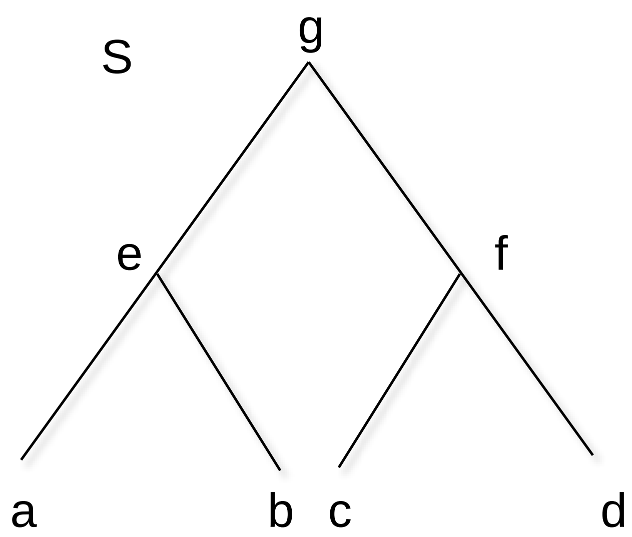
# Polytomies

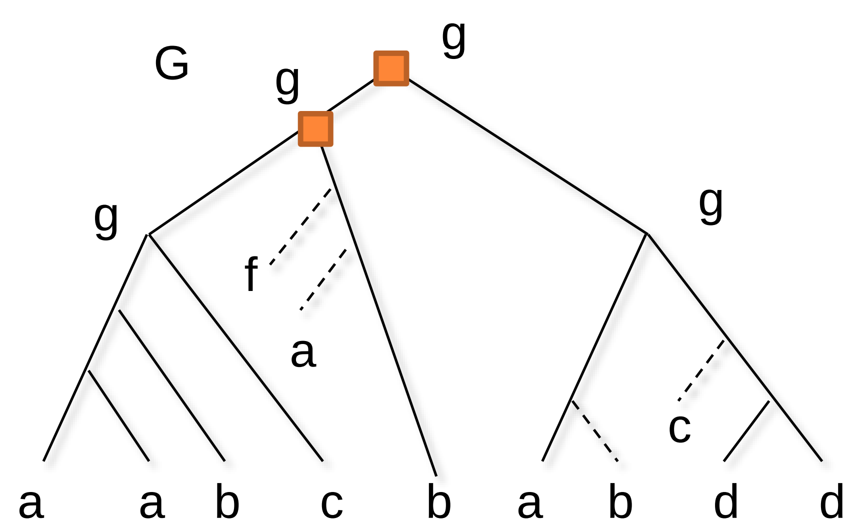
- Each polytomy can be solved independently *(Chang & Eulenstein, 2006)*

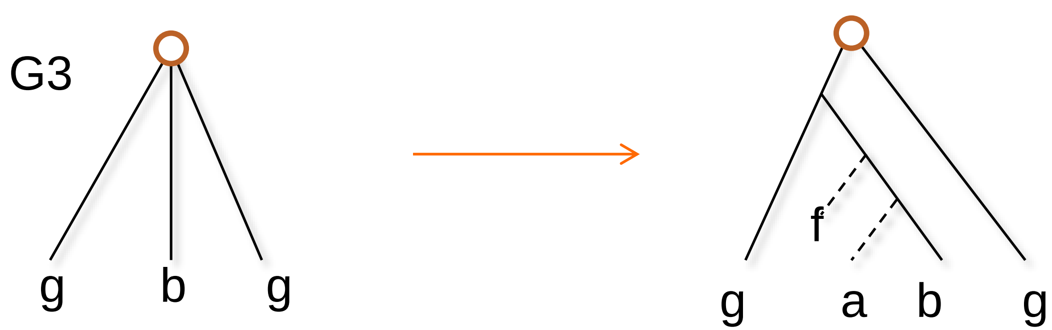


# Polytomies

- Each polytomy can be solved independently *(Chang & Eulenstein, 2006)*

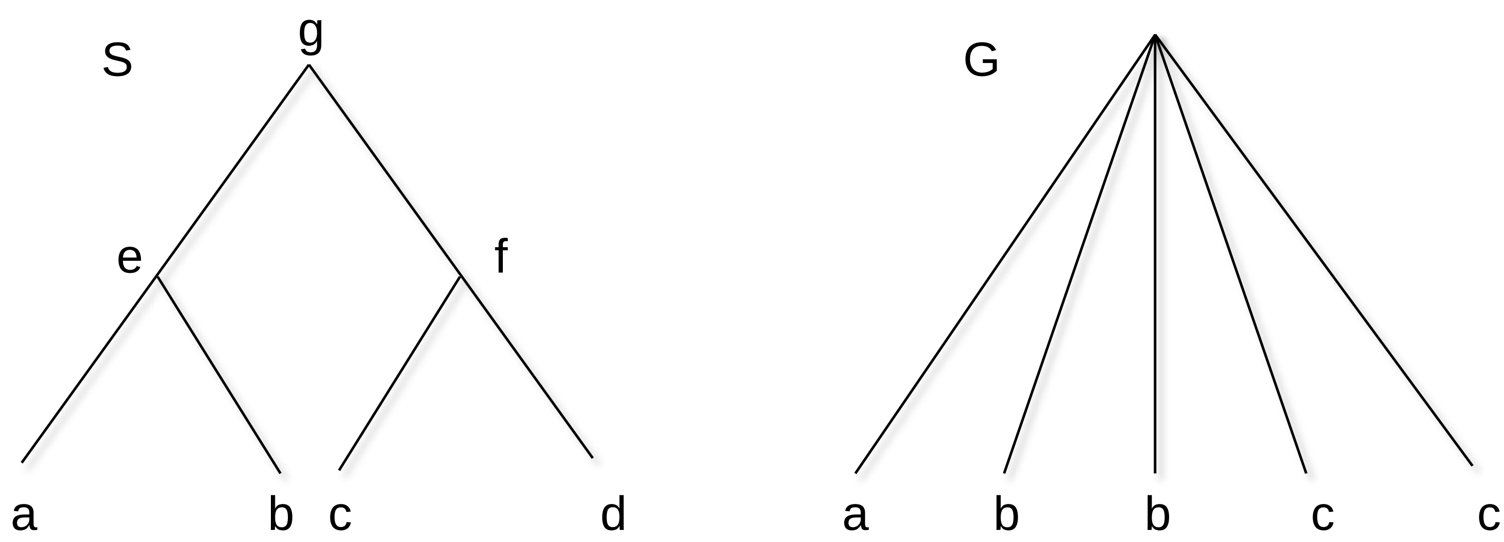


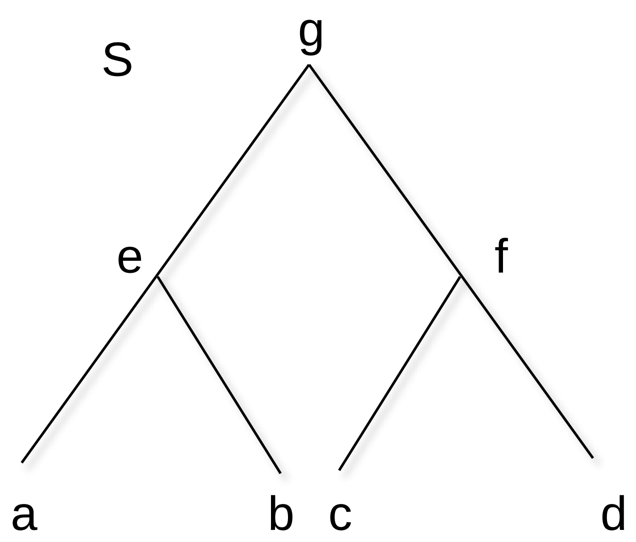
# The core problem

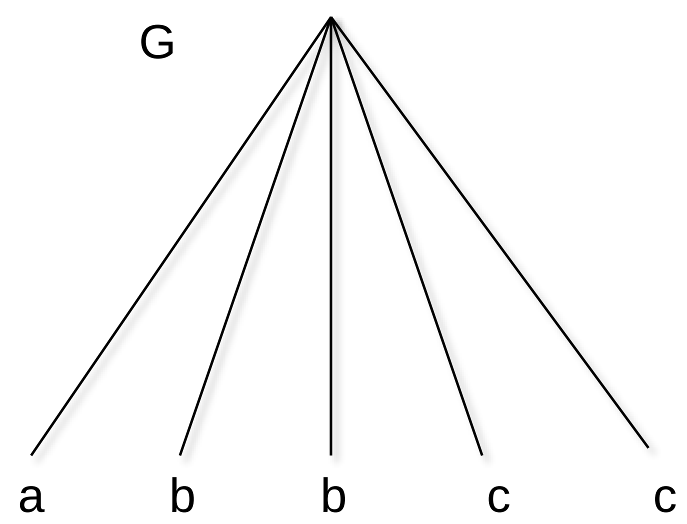
- Find the minimum cost reconciliation between a species tree and a polytomy



# Resolution

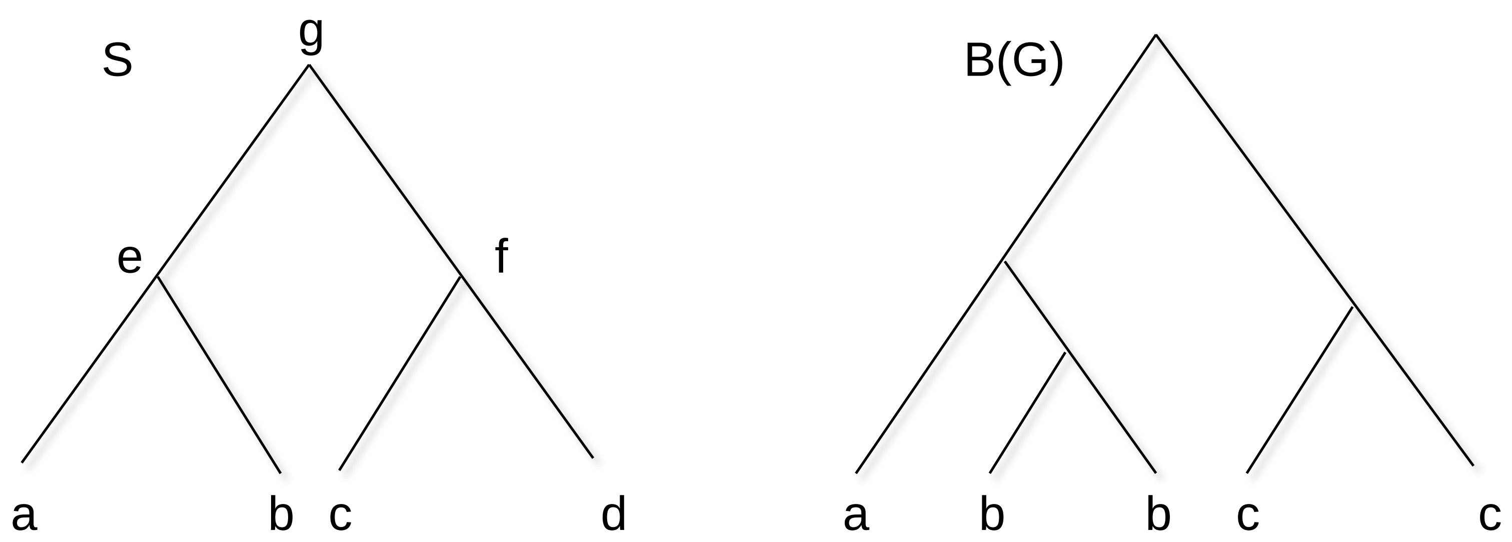
- A reconciliation between S and a **binary refinement** of G.



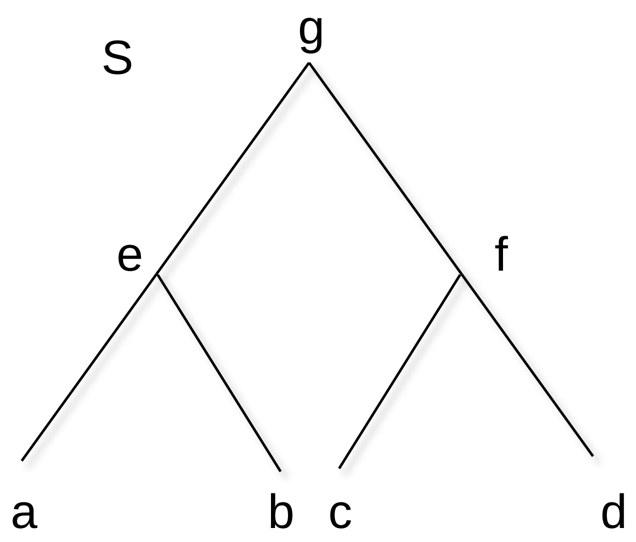
# Resolution

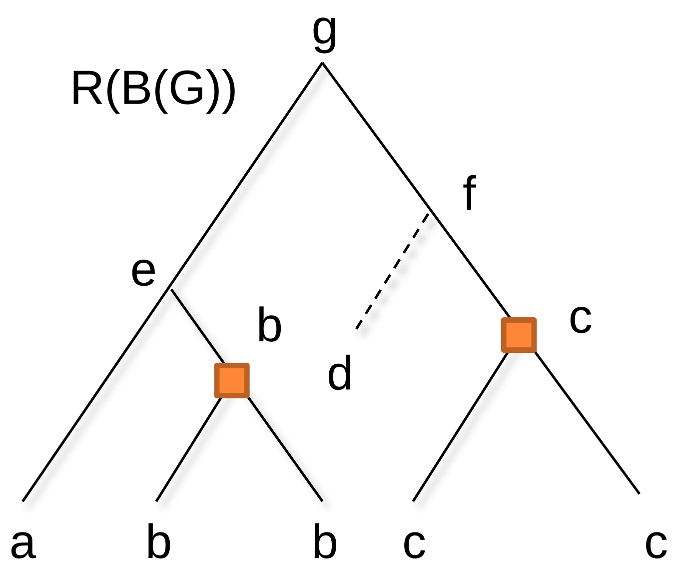
- B(G) is a binary refinement of G



# Resolution

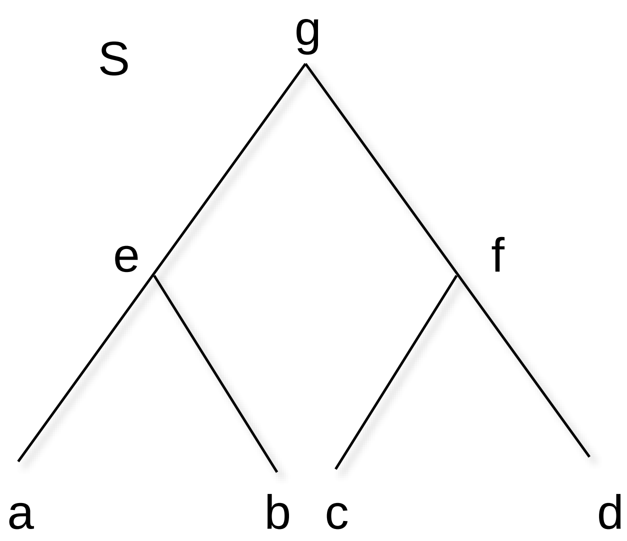
- R(B(G)) is a reconciliation between S and B(G)

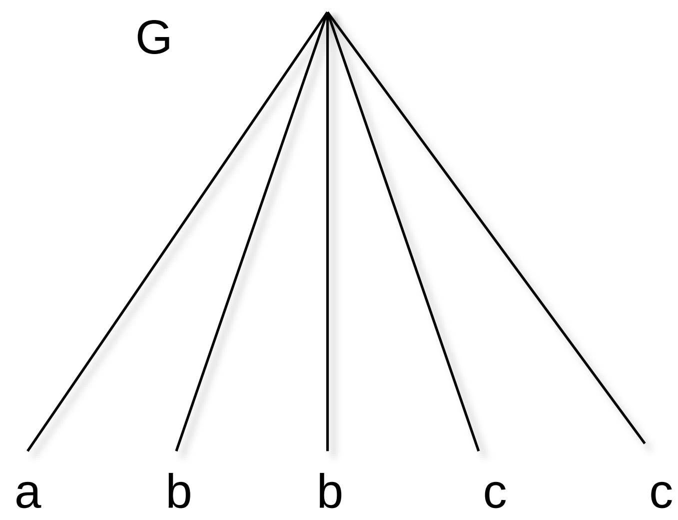


# Problem statement

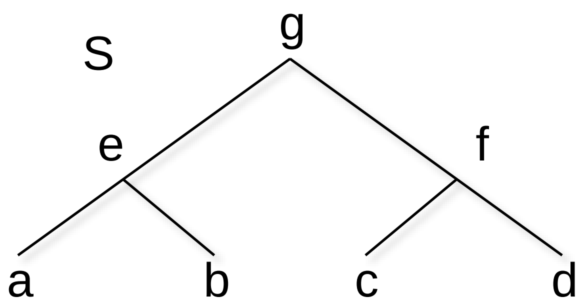
- **Given** : a binary species tree S and a polytomy G
- **Find** : a minimum mutation cost resolution of G.

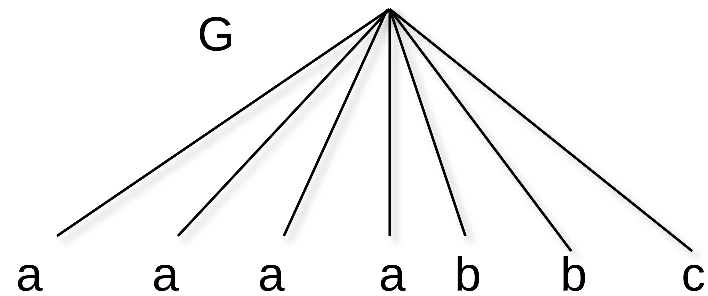


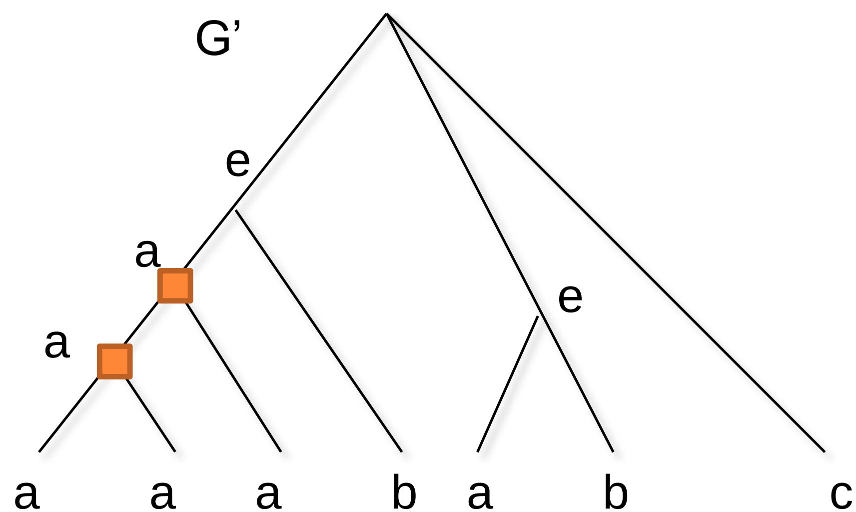



# Partial resolution at node s

- A tree obtained from G in which every subtree rooted at a node labeled s is consistent with the species tree.
- Every descendant of s is part of one of these subtrees.

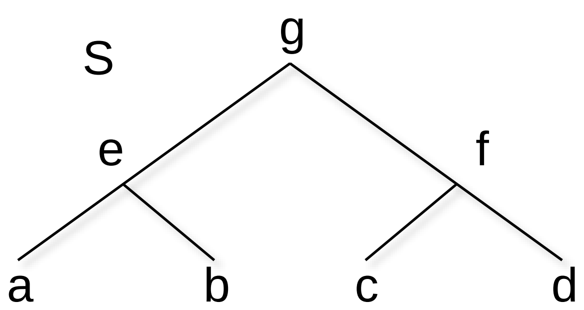


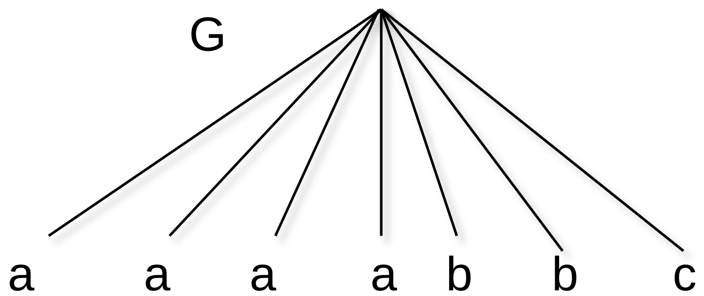



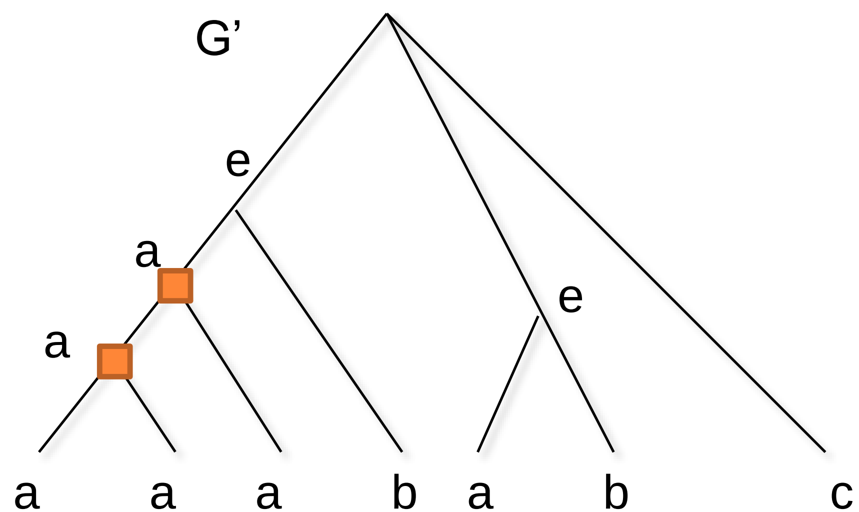



# Partial resolution cost

- The mutation cost of a partial resolution is the sum of the costs of all of its subtrees

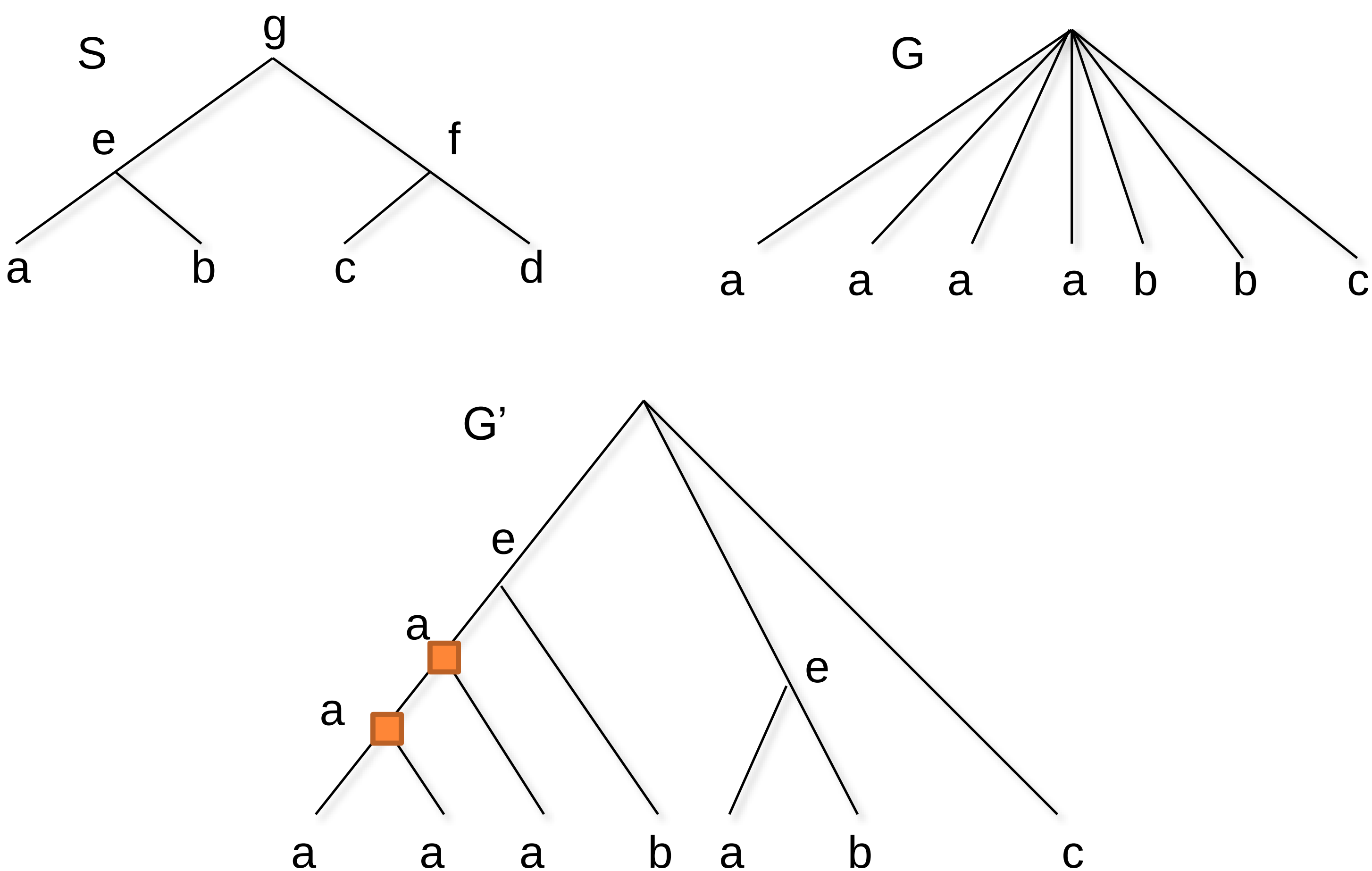


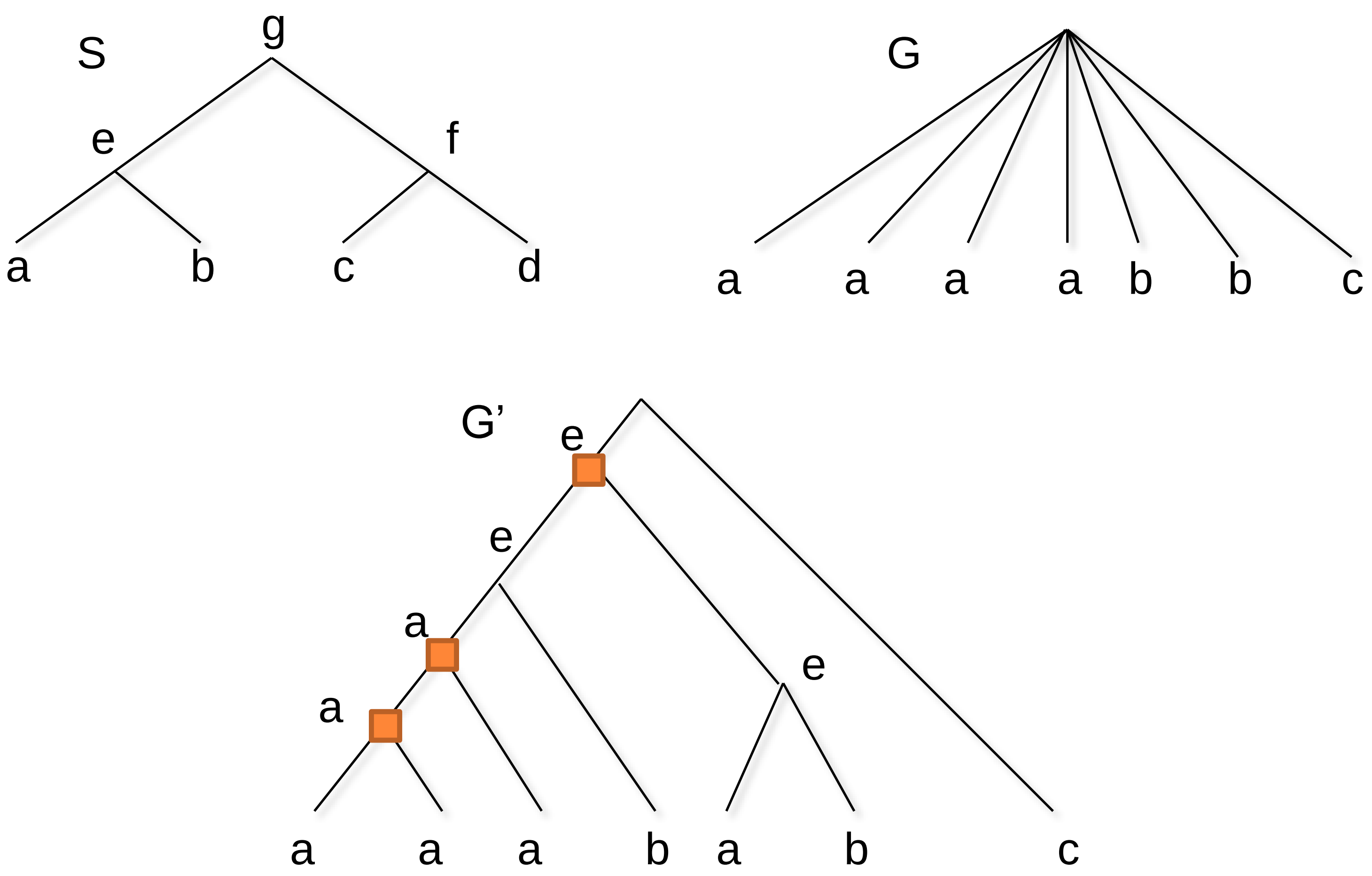
# k-partial resolution at node s

- A partial resolution with exactly k maximal subtrees rooted at s.



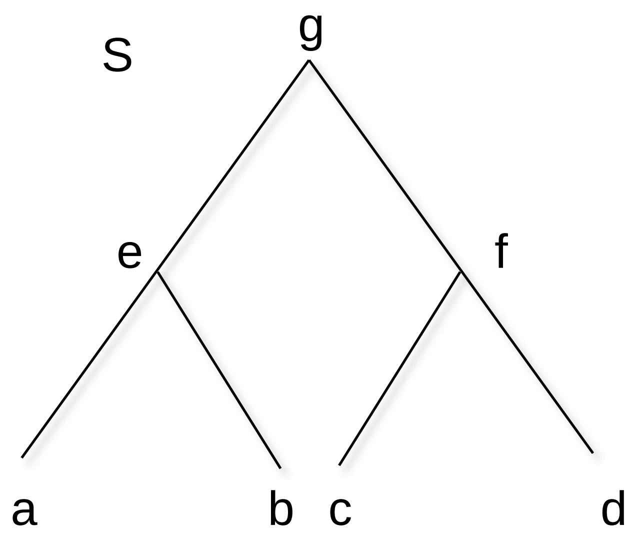
# k-partial resolution at node s

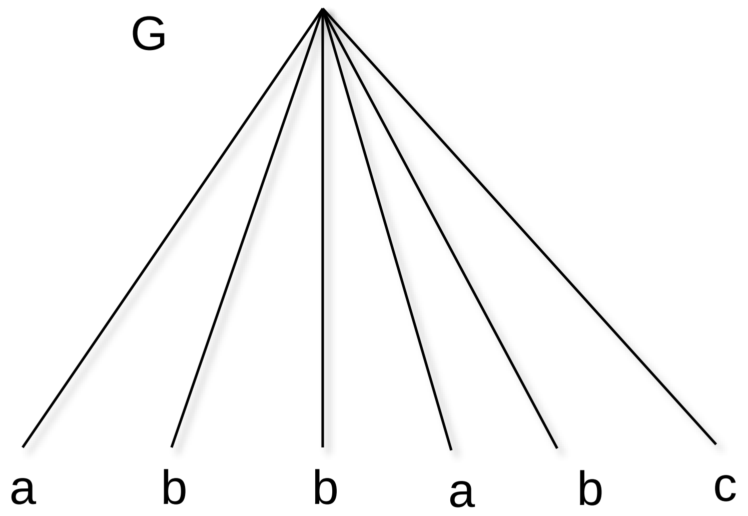
- A partial resolution with exactly k maximal subtrees rooted at s.



# Methodology

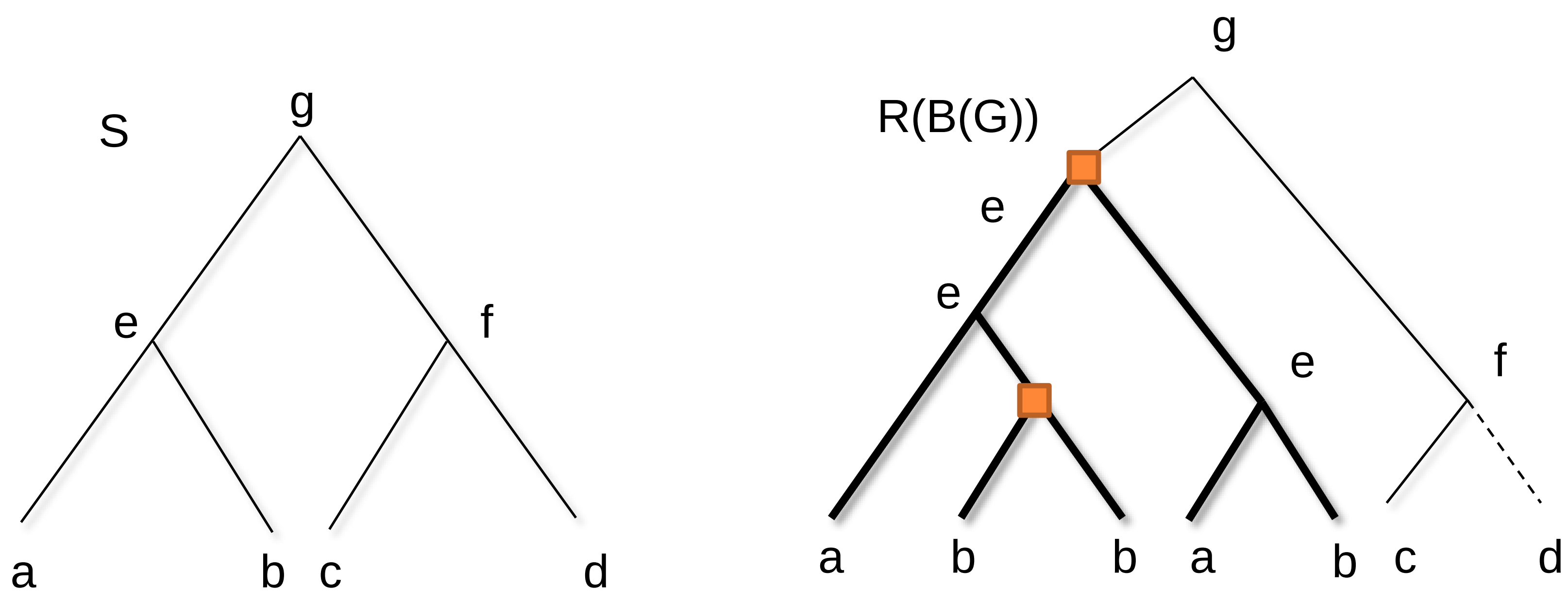
- Idea : an optimal resolution contains a minimum k-partial resolution at s, for every node s in V(S)



# Methodology

- R(B(G)) has a 1-partial resolution at e
- It also has a 2-partial resolution at e



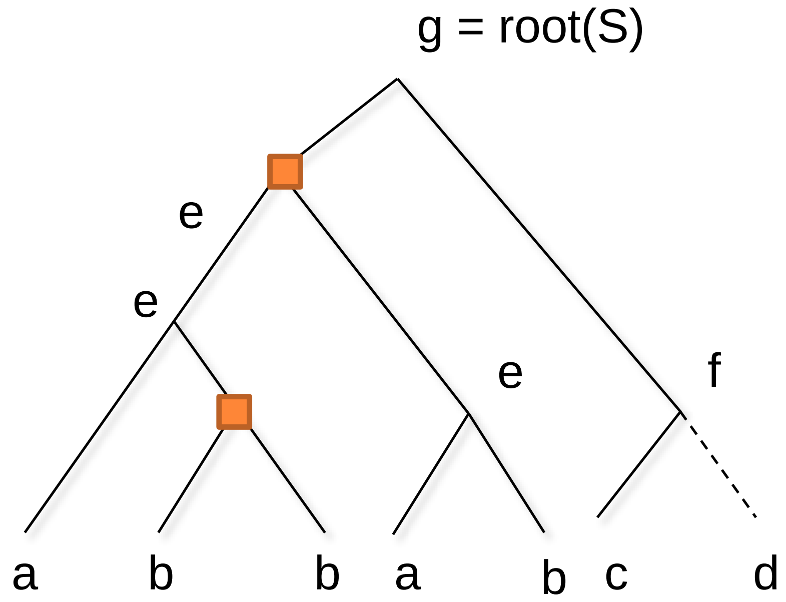

- For which k's does the optimal resolution contain a k-partial resolution ?

# Methodology

- M(s, k) denotes the minimum cost of a k-partial resolution at s

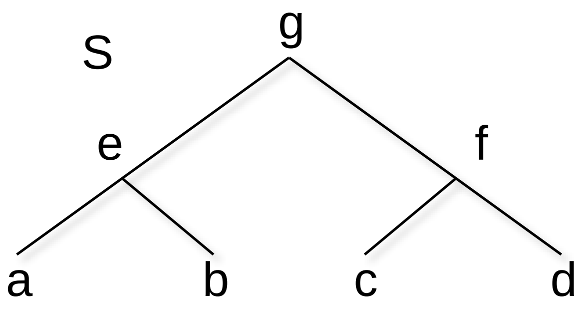
- M(root(S), 1) is the minimum cost of the full resolution of G
  - The solution is a 1-partial resolution at root(S)

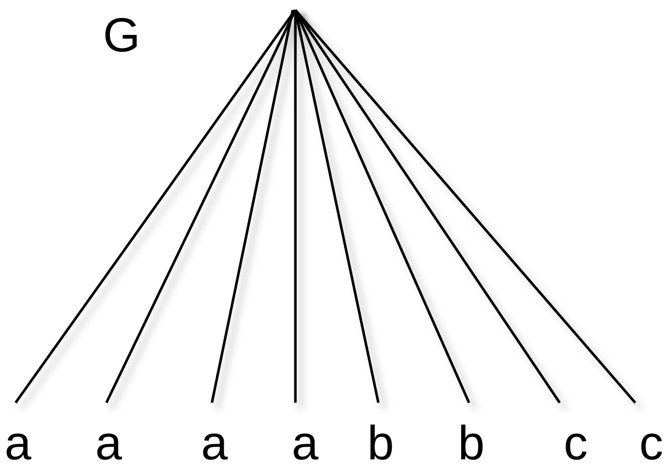
R(B(G)) : a 1-partial resolution at g



# Computation of M(s, k)

- We compute the values of M(s, k) for each node s in V(S) in a bottom-up manner, and for every k.

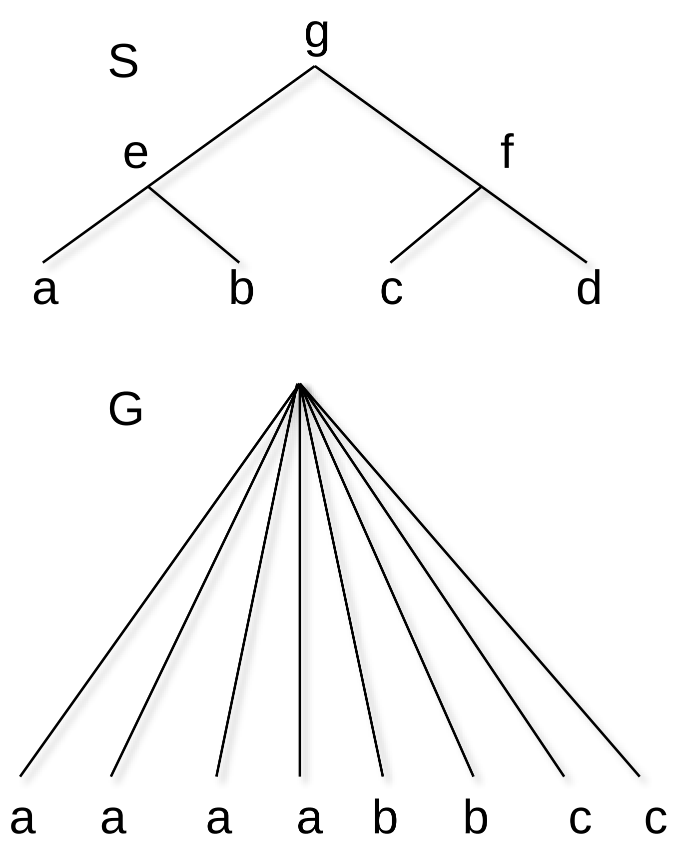



| k = | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| M(a, k) | | | | | | |
| M(b, k) | | | | | | |
| M(c, k) | | | | | | |
| M(d, k) | | | | | | |
| M(f, k) | | | | | | |
| M(e, k) | | | | | | |
| M(g, k) | | | | | | |

# Computation of M(s, k)

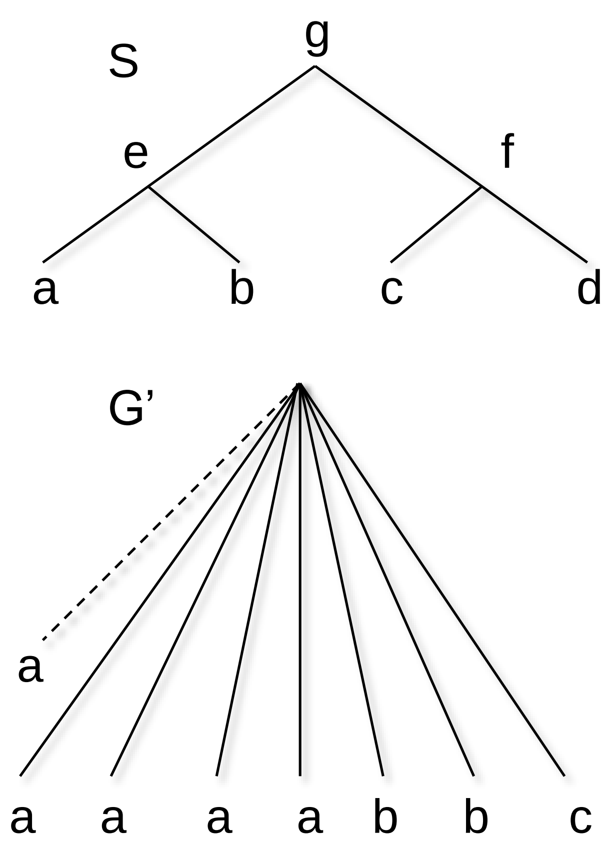
- M(a, 4) = 0




| k = | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| M(a, k) | | | | 0 | | |
| M(b, k) | | | | | | |
| M(c, k) | | | | | | |
| M(d, k) | | | | | | |
| M(f, k) | | | | | | |
| M(e, k) | | | | | | |
| M(g, k) | | | | | | |

# Computation of M(s, k)

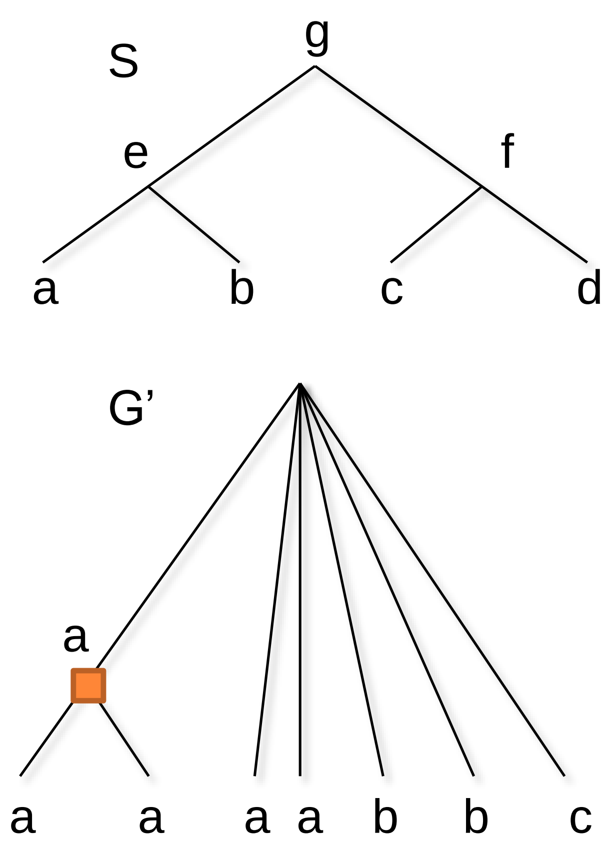
- M(a, 5) = 1 (one loss in a)




| k = | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| M(a, k) | | | | 0 | 1 | |
| M(b, k) | | | | | | |
| M(c, k) | | | | | | |
| M(d, k) | | | | | | |
| M(e, k) | | | | | | |
| M(f, k) | | | | | | |
| M(g, k) | | | | | | |

# Computation of M(s, k)

- M(a, 3) = 1 (one duplication in a)

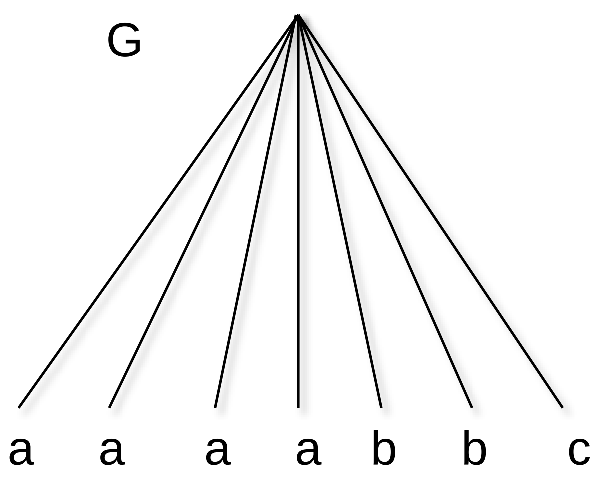



| k = | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| M(a, k) | | | 1 | 0 | 1 | |
| M(b, k) | | | | | | |
| M(c, k) | | | | | | |
| M(d, k) | | | | | | |
| M(e, k) | | | | | | |
| M(f, k) | | | | | | |
| M(g, k) | | | | | | |

# Computation of M(s, k)

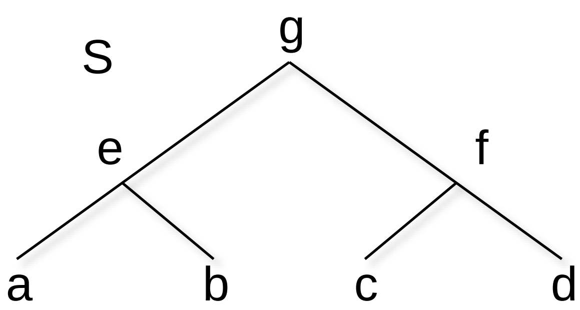
- Let nb(s) denote the number of leaves of G labeled s
  - For instance, nb(a) = 4, nb(b) = 2, …

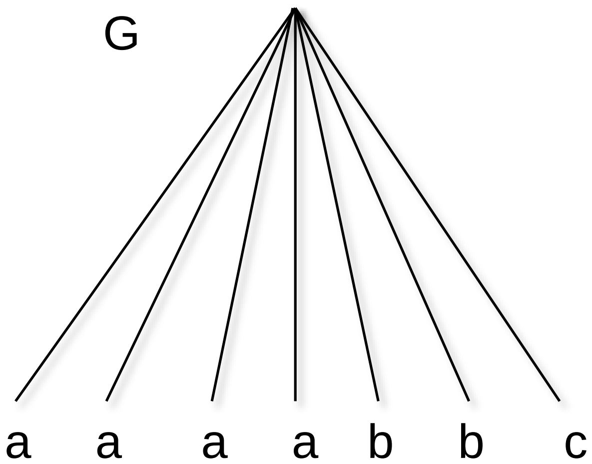
- In general, if s is a leaf, then M(s, k) = |k - nb(s)|



# Computation of M(s, k)

- The leaf values are easy to compute
- M(s, k) = |k – nb(s)|

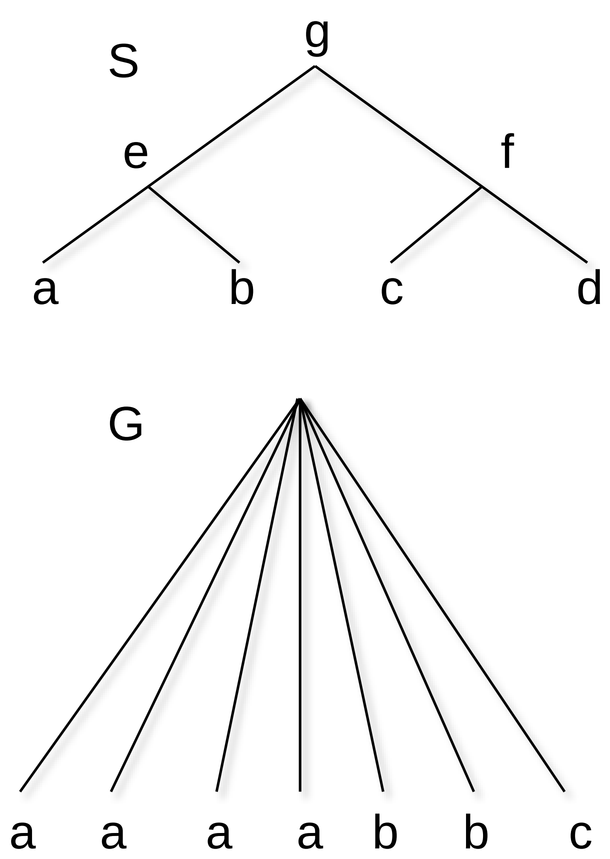



| k = | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| M(a, k) | 3 | 2 | 1 | **0** | 1 | 2 |
| M(b, k) | 1 | **0** | 1 | 2 | 3 | 4 |
| M(c, k) | **0** | 1 | 2 | 3 | 4 | 5 |
| M(d, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(e, k) | | | | | | |
| M(f, k) | | | | | | |
| M(g, k) | | | | | | |

# Computation of M(s, k)

- Computing M(e, k)




| k = | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| M(a, k) | 3 | 2 | 1 | 0 | 1 | 2 |
| M(b, k) | 1 | 0 | 1 | 2 | 3 | 4 |
| M(c, k) | 0 | 1 | 2 | 3 | 4 | 5 |
| M(d, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(e, k) | | | | | | |

# Computation of M(s, k)

- Either
  - M(e, 2) = M(a, 2) + M(b, 2)   *(from above – indicates speciation)*
  - M(e, 2) = M(e, 1) + 1   *(from the left – indicates a loss)*
  - M(e, 2) = M(e, 1) + 1   *(from the left – indicates a duplication)*

| k = | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| M(a, k) | 3 | 2 | 1 | 0 | 1 | 2 |
| M(b, k) | 1 | 0 | 1 | 2 | 3 | 4 |
| M(c, k) | 0 | 1 | 2 | 3 | 4 | 5 |
| M(d, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(e, k) | x | | y | z | | |

+

**+1 loss**   **+1 dup**

# Computation of M(s, k)

- Temporarily let M(s, k) = M(s1, k) + M(s2, k) for every k

| k = | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| M(a, k) | 3 | 2 | 1 | 0 | 1 | 2 |
| M(b, k) | 1 | 0 | 1 | 2 | 3 | 4 |
| M(c, k) | 0 | 1 | 2 | 3 | 4 | 5 |
| M(d, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(e, k) | 4 | 2 | 2 | 2 | 4 | 6 |

# Computation of M(s, k)

- Keep the minimum values only
  - If there are more than one, they will be grouped together

| k = | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| M(a, k) | 3 | 2 | 1 | 0 | 1 | 2 |
| M(b, k) | 1 | 0 | 1 | 2 | 3 | 4 |
| M(c, k) | 0 | 1 | 2 | 3 | 4 | 5 |
| M(d, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(e, k) | | 2 | 2 | 2 | | |

# Computation of M(s, k)

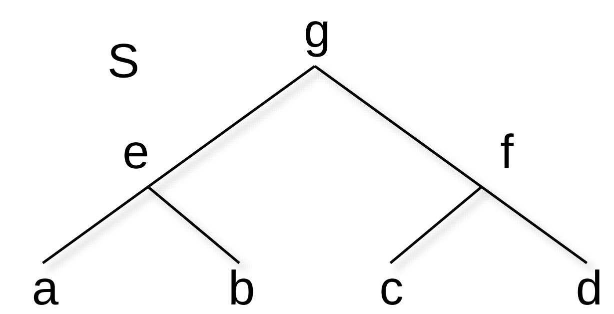
- Extend the minimums, adding one for each cell traversed

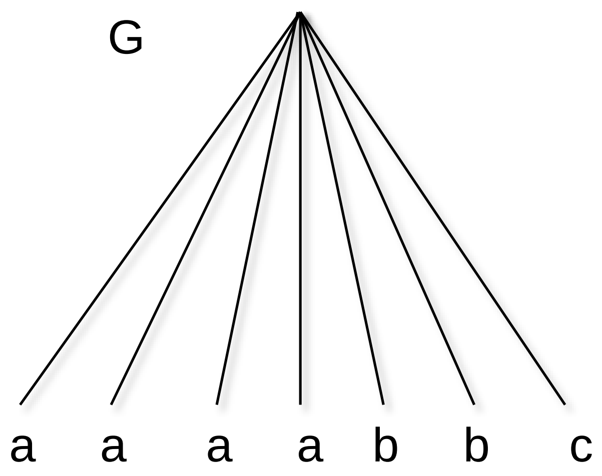
| k = | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| M(a, k) | 3 | 2 | 1 | 0 | 1 | 2 |
| M(b, k) | 1 | 0 | 1 | 2 | 3 | 4 |
| M(c, k) | 0 | 1 | 2 | 3 | 4 | 5 |
| M(d, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(e, k) | 3 | 2 | 2 | 2 | 3 | 4 |

**+1** **+1** **+1**

# Computation of M(s, k)

- The whole table can be filled this way

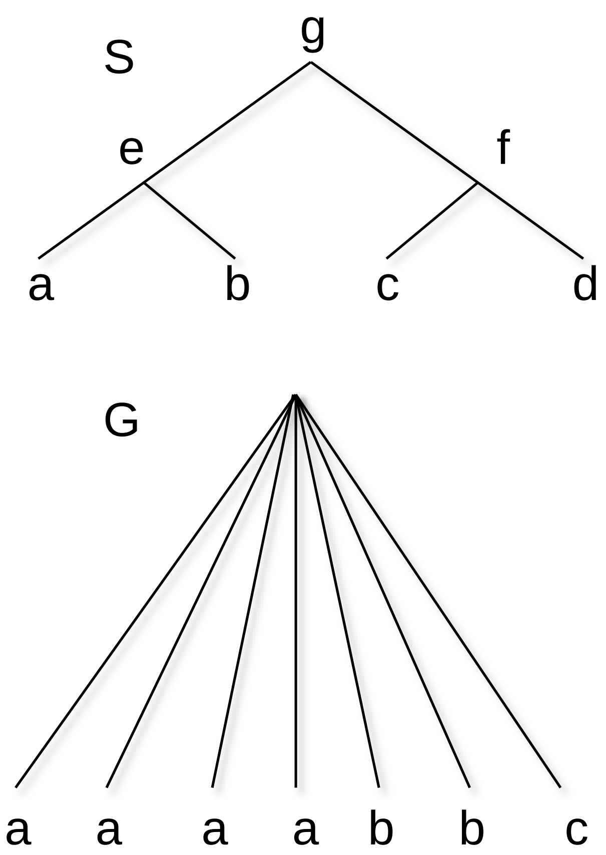



| k = | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| M(a, k) | 3 | 2 | 1 | 0 | 1 | 2 |
| M(b, k) | 1 | 0 | 1 | 2 | 3 | 4 |
| M(c, k) | 0 | 1 | 2 | 3 | 4 | 5 |
| M(d, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(e, k) | 3 | 2 | 2 | 2 | 3 | 4 |
| M(f, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(g, k) | 4 | 4 | 5 | 6 | 7 | 8 |

# Computation of M(s, k)

- The minimum cost of a resolution of G is M(g, 1) = 4




| k = | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| M(a, k) | 3 | 2 | 1 | 0 | 1 | 2 |
| M(b, k) | 1 | 0 | 1 | 2 | 3 | 4 |
| M(c, k) | 0 | 1 | 2 | 3 | 4 | 5 |
| M(d, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(e, k) | 3 | 2 | 2 | 2 | 3 | 4 |
| M(f, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(g, k) | 4 | 4 | 5 | 6 | 7 | 8 |

# Building the resolution

- Using the table, we'll find the number of duplications and losses for each node of s.

| k = | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| M(a, k) | 3 | 2 | 1 | 0 | 1 | 2 |
| M(b, k) | 1 | 0 | 1 | 2 | 3 | 4 |
| M(c, k) | 0 | 1 | 2 | 3 | 4 | 5 |
| M(d, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(e, k) | 3 | 2 | 2 | 2 | 3 | 4 |
| M(f, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(g, k) | 4 | 4 | 5 | 6 | 7 | 8 |

# Building the resolution

- Backtrack where the value of M(g, 1) came from

| k = | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| M(a, k) | 3 | 2 | 1 | 0 | 1 | 2 |
| M(b, k) | 1 | 0 | 1 | 2 | 3 | 4 |
| M(c, k) | 0 | 1 | 2 | 3 | 4 | 5 |
| M(d, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(e, k) | 3 | 2 | 2 | 2 | 3 | 4 |
| M(f, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(g, k) | 4 | 4 | 5 | 6 | 7 | 8 |

# Building the resolution

- Backtrack where the value of M(g, 1) came from
  - M(g, 1) = M(e, 1) + M(f, 1)

| k = | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| M(a, k) | 3 | 2 | 1 | 0 | 1 | 2 |
| M(b, k) | 1 | 0 | 1 | 2 | 3 | 4 |
| M(c, k) | 0 | 1 | 2 | 3 | 4 | 5 |
| M(d, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(e, k) | 3 | 2 | 2 | 2 | 3 | 4 |
| M(f, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(g, k) | 4 | 4 | 5 | 6 | 7 | 8 |

# Building the resolution

- Backtrack where the value of M(g, 1) came from
  - M(f, 1) = M(c, 1) + M(d, 1)

| k = | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| M(a, k) | 3 | 2 | 1 | 0 | 1 | 2 |
| M(b, k) | 1 | 0 | 1 | 2 | 3 | 4 |
| M(c, k) | 0 | 1 | 2 | 3 | 4 | 5 |
| M(d, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(e, k) | 3 | 2 | 2 | 2 | 3 | 4 |
| M(f, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(g, k) | 4 | 4 | 5 | 6 | 7 | 8 |

# Building the resolution

- Backtrack where the value of M(g, 1) came from
  - M(e, 1) = M(e, 2) + 1

| k = | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| M(a, k) | 3 | 2 | 1 | 0 | 1 | 2 |
| M(b, k) | 1 | 0 | 1 | 2 | 3 | 4 |
| M(c, k) | 0 | 1 | 2 | 3 | 4 | 5 |
| M(d, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(e, k) | 3 | 2 | 2 | 2 | 3 | 4 |
| M(f, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(g, k) | 4 | 4 | 5 | 6 | 7 | 8 |

- One duplication in e !

# Building the resolution

- Backtrack where the value of M(g, 1) came from
  - M(e, 2) = M(a, 2) + M(b, 2)

| k = | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| M(a, k) | 3 | 2 | 1 | 0 | 1 | 2 |
| M(b, k) | 1 | 0 | 1 | 2 | 3 | 4 |
| M(c, k) | 0 | 1 | 2 | 3 | 4 | 5 |
| M(d, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(e, k) | 3 | 2 | 2 | 2 | 3 | 4 |
| M(f, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(g, k) | 4 | 4 | 5 | 6 | 7 | 8 |

# Building the resolution

- For leaves, go to the cell with value zero

| k = | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| M(a, k) | 3 | 2 | 1 | 0 | 1 | 2 |
| M(b, k) | 1 | 0 | 1 | 2 | 3 | 4 |
| M(c, k) | 0 | 1 | 2 | 3 | 4 | 5 |
| M(d, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(e, k) | 3 | 2 | 2 | 2 | 3 | 4 |
| M(f, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(g, k) | 4 | 4 | 5 | 6 | 7 | 8 |

- Two duplications in a !

# Building the resolution

- For leaves, go to the cell with value zero

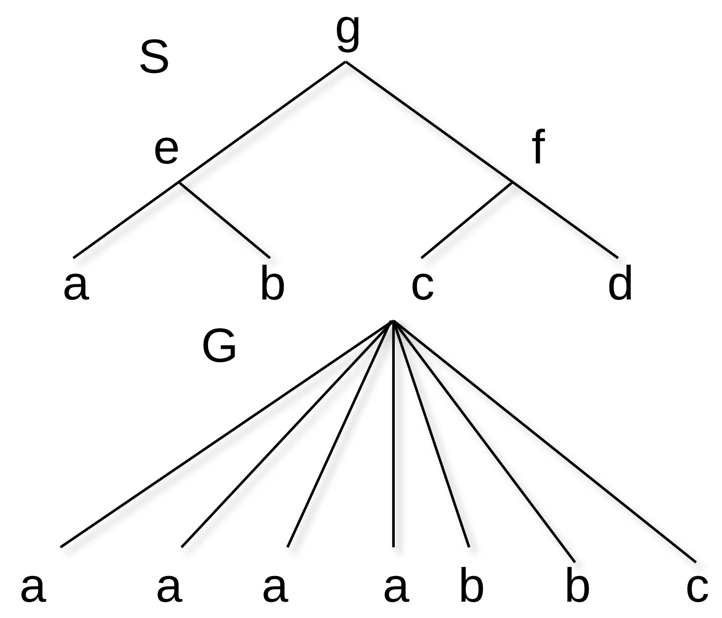
| k = | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| M(a, k) | 3 | 2 | 1 | 0 | 1 | 2 |
| M(b, k) | 1 | 0 | 1 | 2 | 3 | 4 |
| M(c, k) | 0 | 1 | 2 | 3 | 4 | 5 |
| M(d, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(e, k) | 3 | 2 | 2 | 2 | 3 | 4 |
| M(f, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(g, k) | 4 | 4 | 5 | 6 | 7 | 8 |

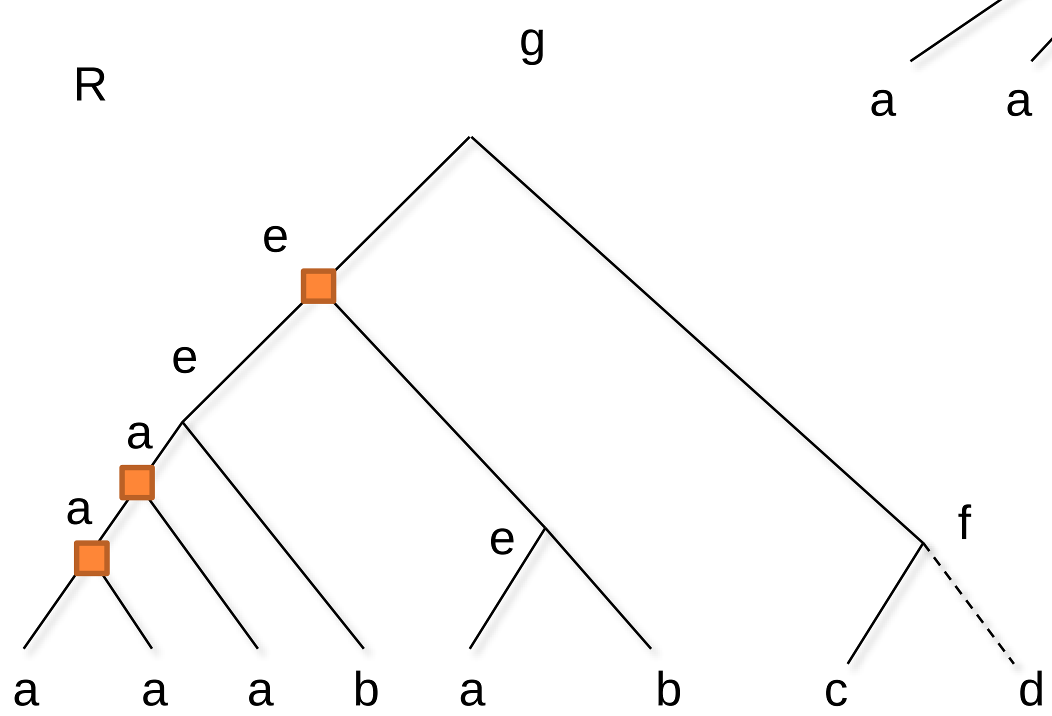
- If there is no zero, assume it is at column 0

- One loss in d

# Building the resolution

- This gives :
  - 1 duplication in e
  - 1 loss in d
  - 2 duplications in a



# Computing the table

- Problem : we stopped at k = 6, but this value was arbitrary
- Who knows when to stop ?

| k = | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| M(a, k) | 3 | 2 | 1 | 0 | 1 | 2 |
| M(b, k) | 1 | 0 | 1 | 2 | 3 | 4 |
| M(c, k) | 0 | 1 | 2 | 3 | 4 | 5 |
| M(d, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(e, k) | 3 | 2 | 2 | 2 | 3 | 4 |
| M(f, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(g, k) | 4 | 4 | 5 | 6 | 7 | 8 |

# Computing the table

- Computing this table takes O(|S|* k-max) steps

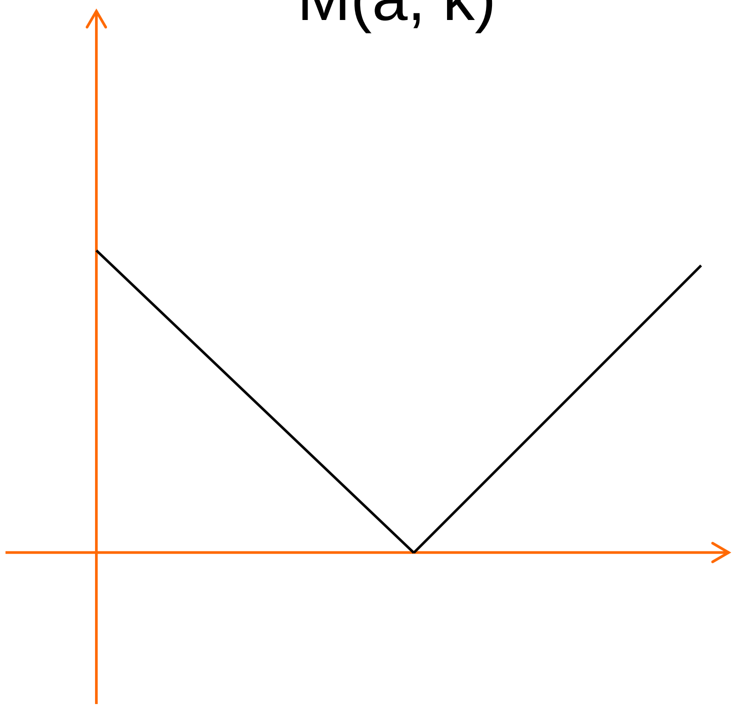
| k = | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| M(a, k) | 3 | 2 | 1 | 0 | 1 | 2 |
| M(b, k) | 1 | 0 | 1 | 2 | 3 | 4 |
| M(c, k) | 0 | 1 | 2 | 3 | 4 | 5 |
| M(d, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(e, k) | 3 | 2 | 2 | 2 | 3 | 4 |
| M(f, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(g, k) | 4 | 4 | 5 | 6 | 7 | 8 |

# Computing the table

- The values of a row follow a pattern

| k = | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| M(a, k) | 3 | 2 | 1 | 0 | 1 | 2 |
| M(b, k) | 1 | 0 | 1 | 2 | 3 | 4 |
| M(c, k) | 0 | 1 | 2 | 3 | 4 | 5 |
| M(d, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(e, k) | 3 | 2 | 2 | 2 | 3 | 4 |
| M(f, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(g, k) | 4 | 4 | 5 | 6 | 7 | 8 |

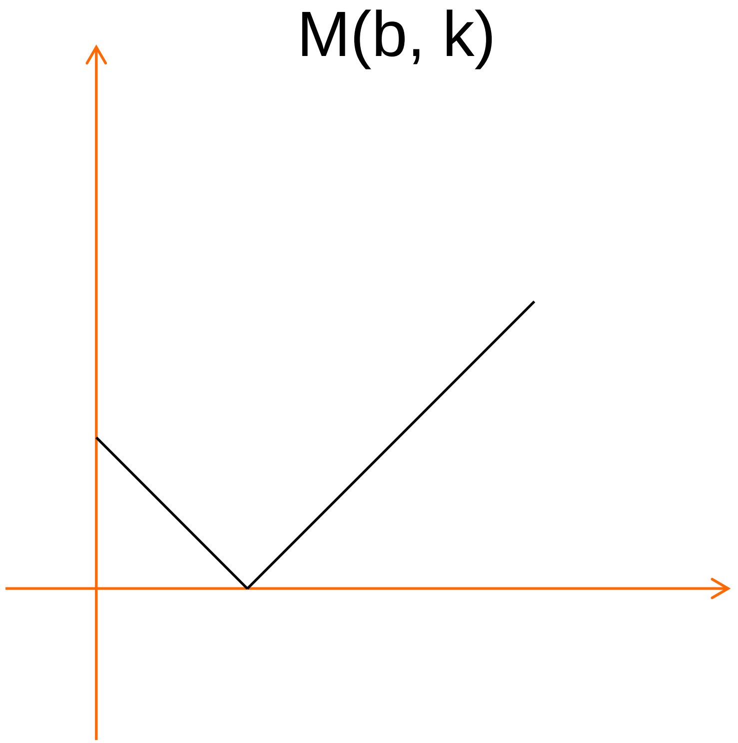
M(a, k)



# Computing the table

- The values of a row follow a pattern

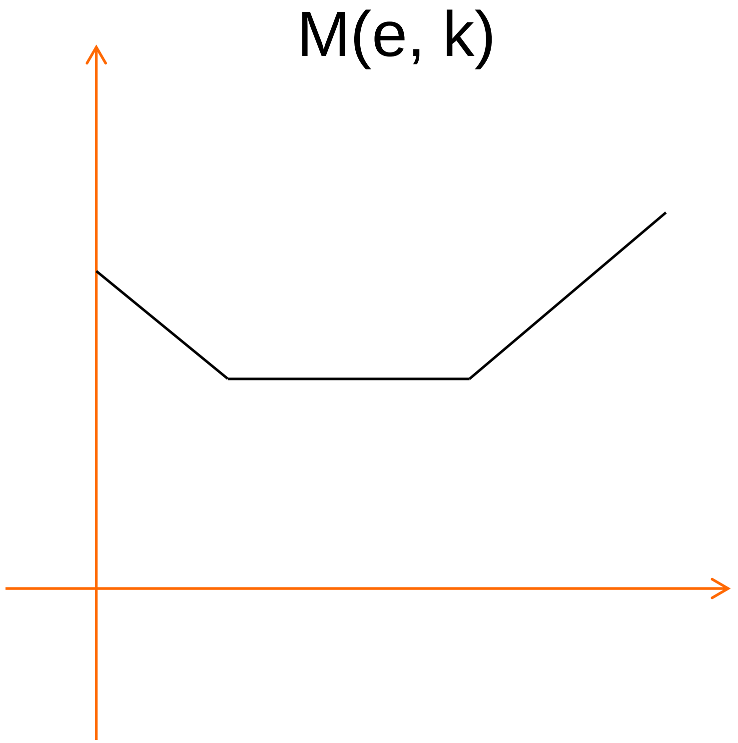
| k = | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| M(a, k) | 3 | 2 | 1 | 0 | 1 | 2 |
| M(b, k) | 1 | 0 | 1 | 2 | 3 | 4 |
| M(c, k) | 0 | 1 | 2 | 3 | 4 | 5 |
| M(d, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(e, k) | 3 | 2 | 2 | 2 | 3 | 4 |
| M(f, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(g, k) | 4 | 4 | 5 | 6 | 7 | 8 |



# Computing the table

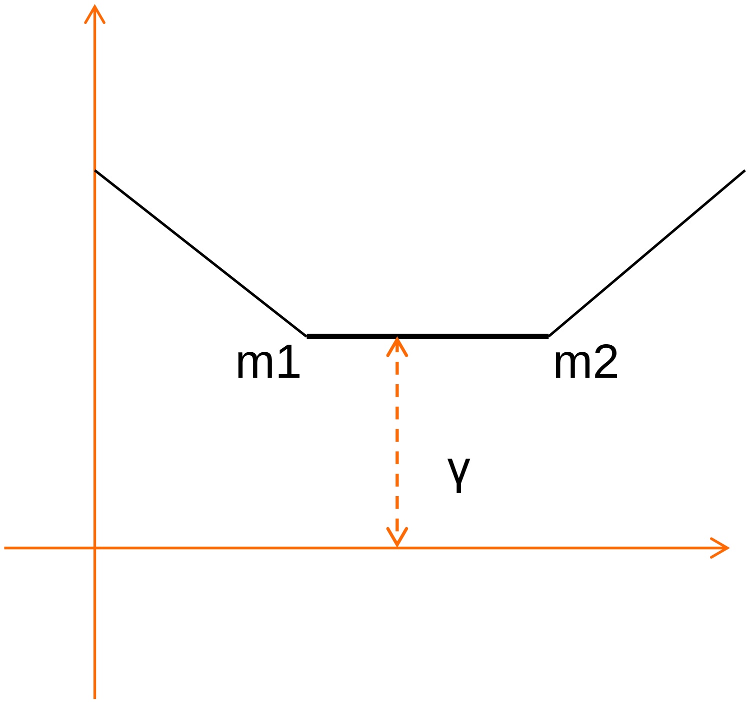
- The values of a row follow a pattern

| k = | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| M(a, k) | 3 | 2 | 1 | 0 | 1 | 2 |
| M(b, k) | 1 | 0 | 1 | 2 | 3 | 4 |
| M(c, k) | 0 | 1 | 2 | 3 | 4 | 5 |
| M(d, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(e, k) | 3 | 2 | 2 | 2 | 3 | 4 |
| M(f, k) | 1 | 2 | 3 | 4 | 5 | 6 |
| M(g, k) | 4 | 4 | 5 | 6 | 7 | 8 |



# Computing the table

- The values of a row follow a pattern

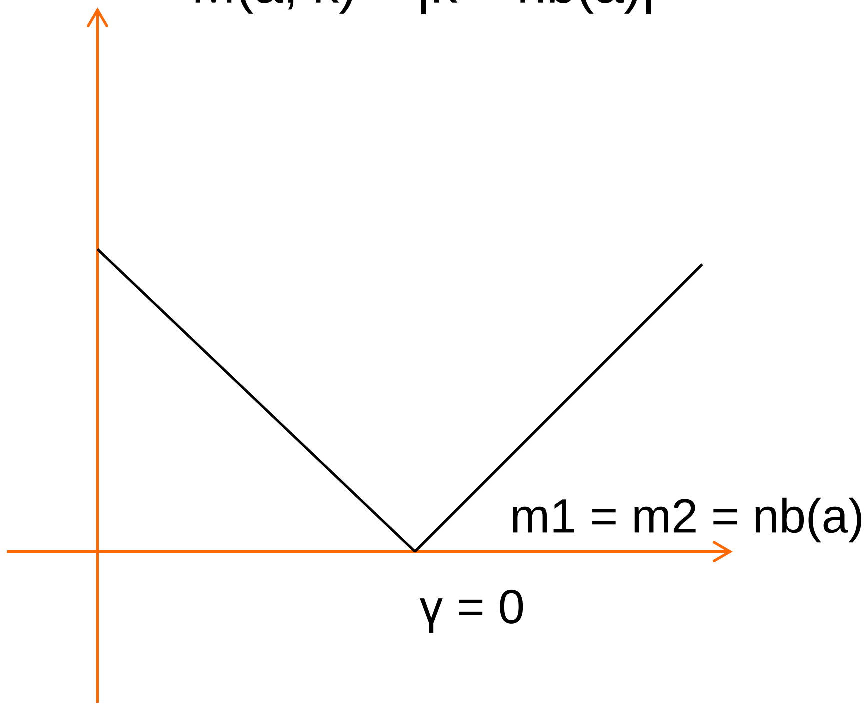



- If we know m1, m2 and γ, we can find the value of M(s, k) for any k in constant time
- m1, m2 are called breakpoints, and γ the minimum value

# Computing the table

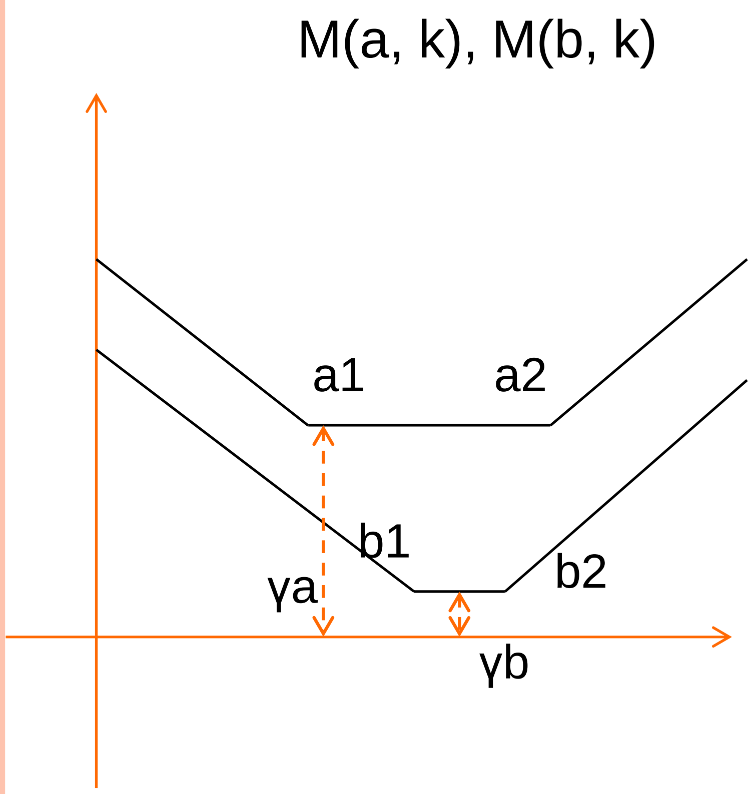
- Finding m1, m2, γ
- Easy for leaf nodes

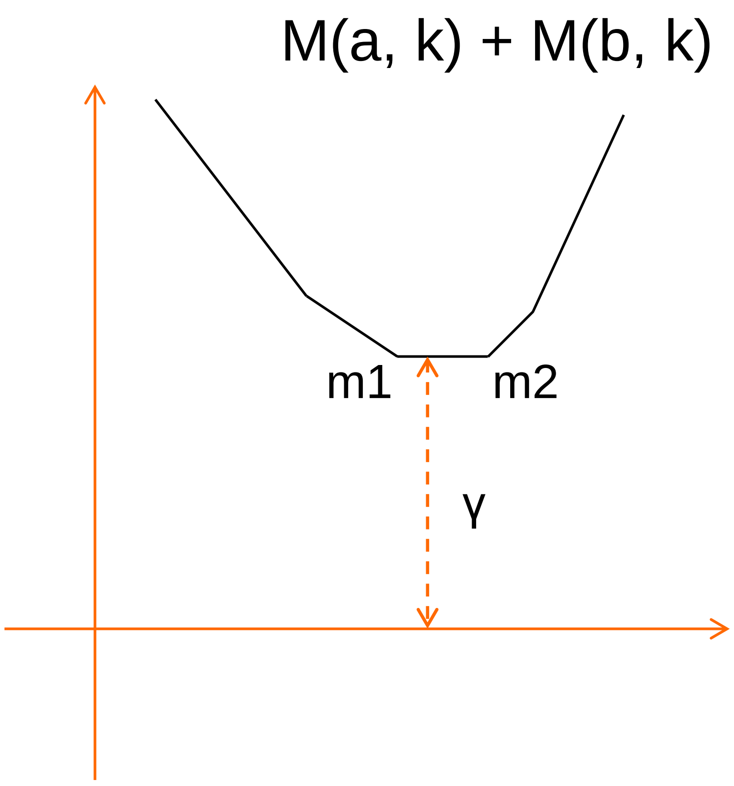
M(a, k) = |k – nb(a)|



# Computing the table

- For an internal node s with children a,b
- The breakpoints and min. val. of M(s, k) can be computed in constant time if we know the breakpoints/min. val. of M(a, k) and M(b, k)



# Conclusion

- Computing one row takes constant time, and there are |S| rows, so the « table » can be computed in O(|S|) steps
- Finding the number of duplications and losses for each node can be done in O(|S|) steps
- Building the resolution can be done in O(|S|) steps as well

# Conclusion

- One polytomy can be solved in O(|S|) steps
- A complete gene tree can have up to |G| polytomies, so a complete resolution can be obtained in O(|G||S|) steps

- In the worst case, a resolution has O(|G||S|) nodes
- Therefore, this algorithm is optimal
  - It runs in as much steps as the maximum size of the output