# Computing Distances Between Evolutionary Trees

Catherine McCartin

Massey University, New Zealand

and

Magnus Bordewich, Michael Hallett, Charles Semple

# Outline

We will consider a series of closely related problems:

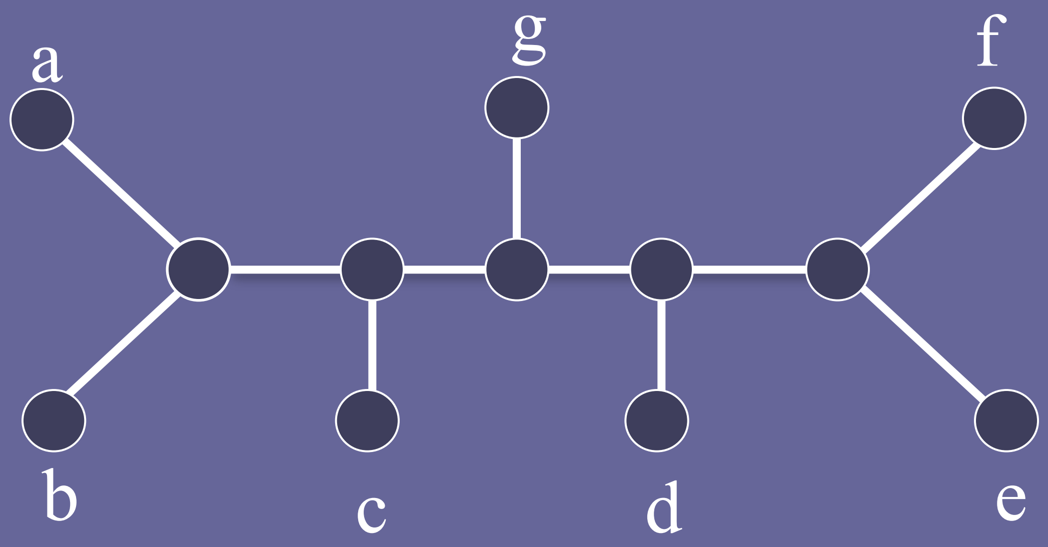
1. TBR distance between pairs of unrooted phlogenetic trees
2. SPR distance between pairs of unrooted phylogenetic trees
3. rSPR distance between pairs of rooted phylogenetic trees

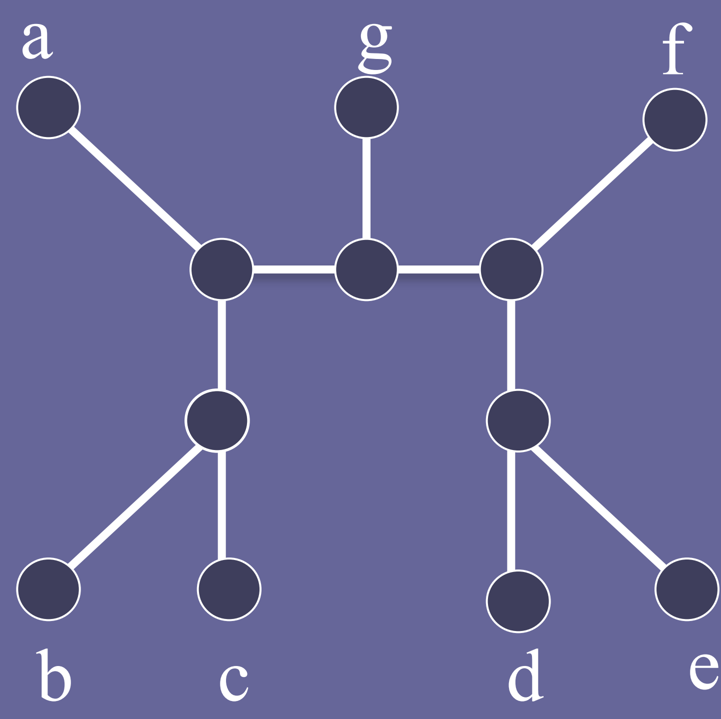
All NP-hard so aim for

1. FPT algorithms of parameterized versions
2. Approximation algorithms

All of the algorithms are variations on a theme.

# unrooted binary phylogenetic X-trees




$X = \{a, b, c, d, e, f, g\}$

# Tree metrics

How ‘far apart’ are these two trees

1. from each other?
2. from the ‘true’ tree?

Is there a true tree?

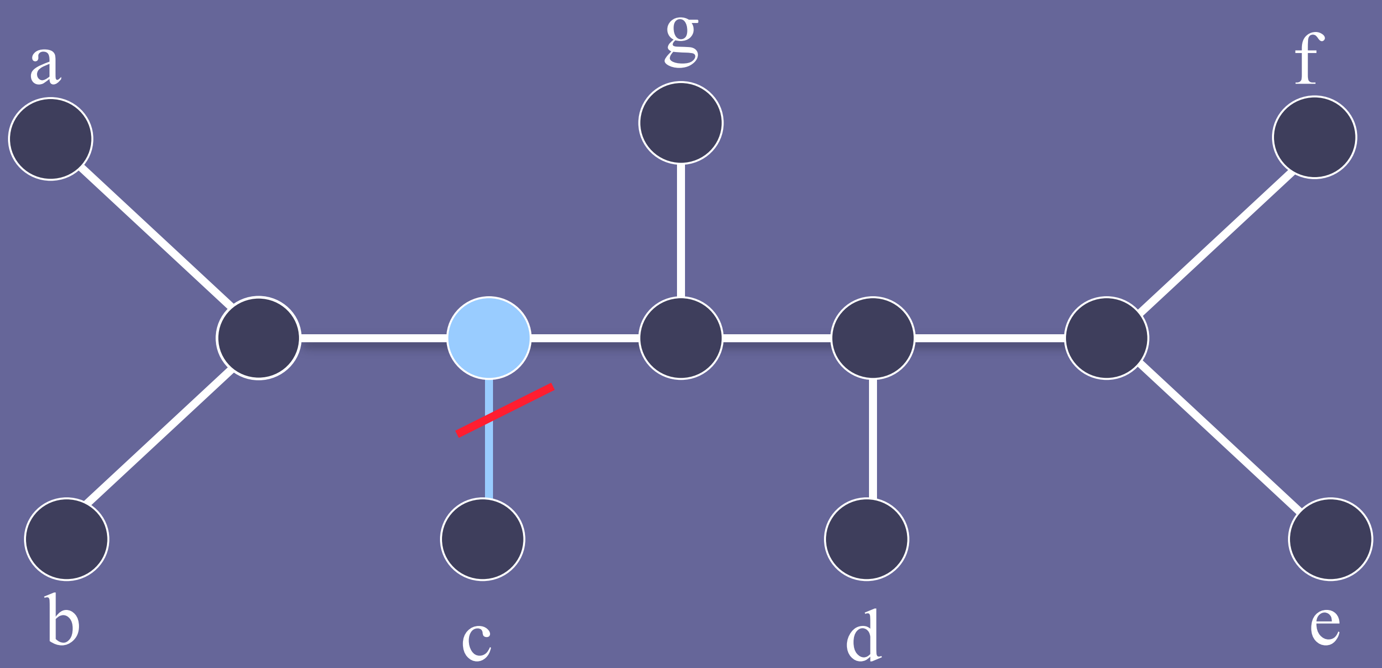
# Tree metrics

How ‘far apart’ are these two trees

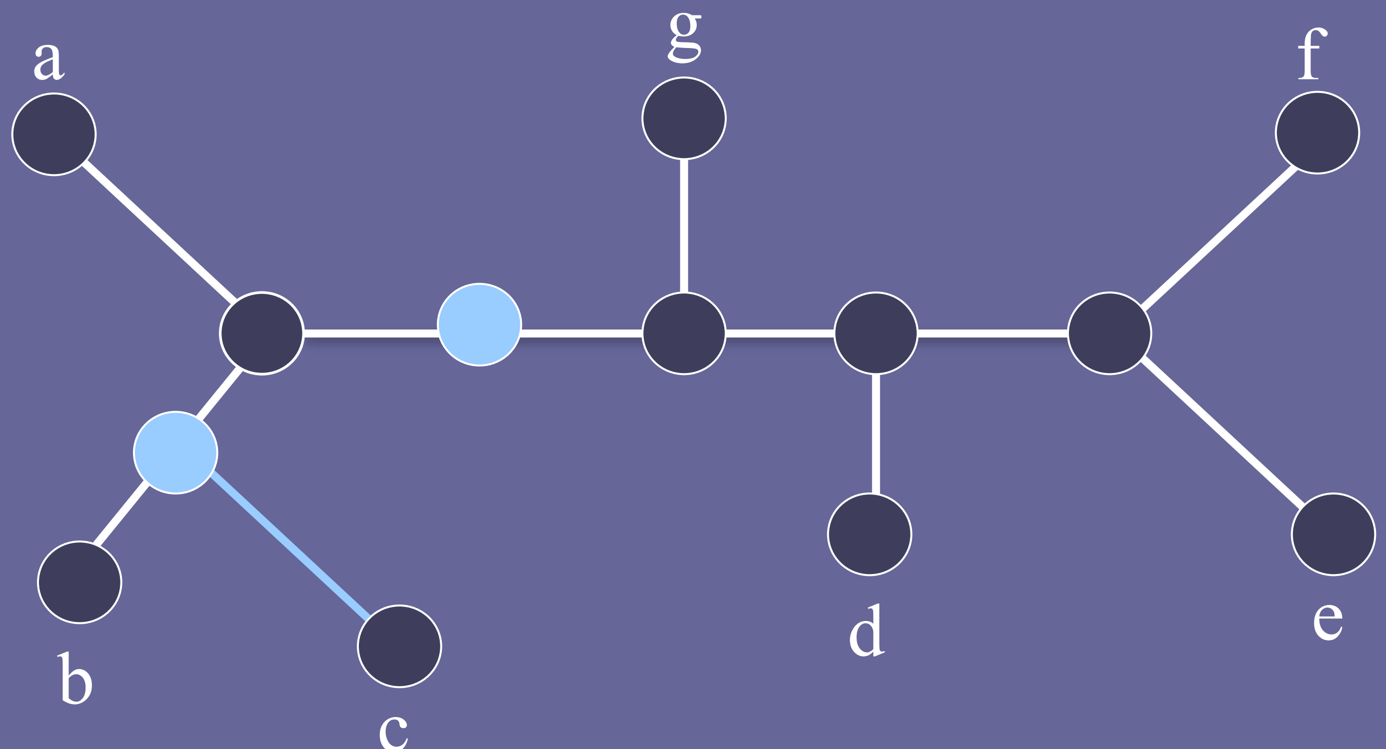
1. from each other?
2. from the ‘true’ tree?

Natural choice is to say two trees are ‘close’ if we can get from one to the other with a few local operations.

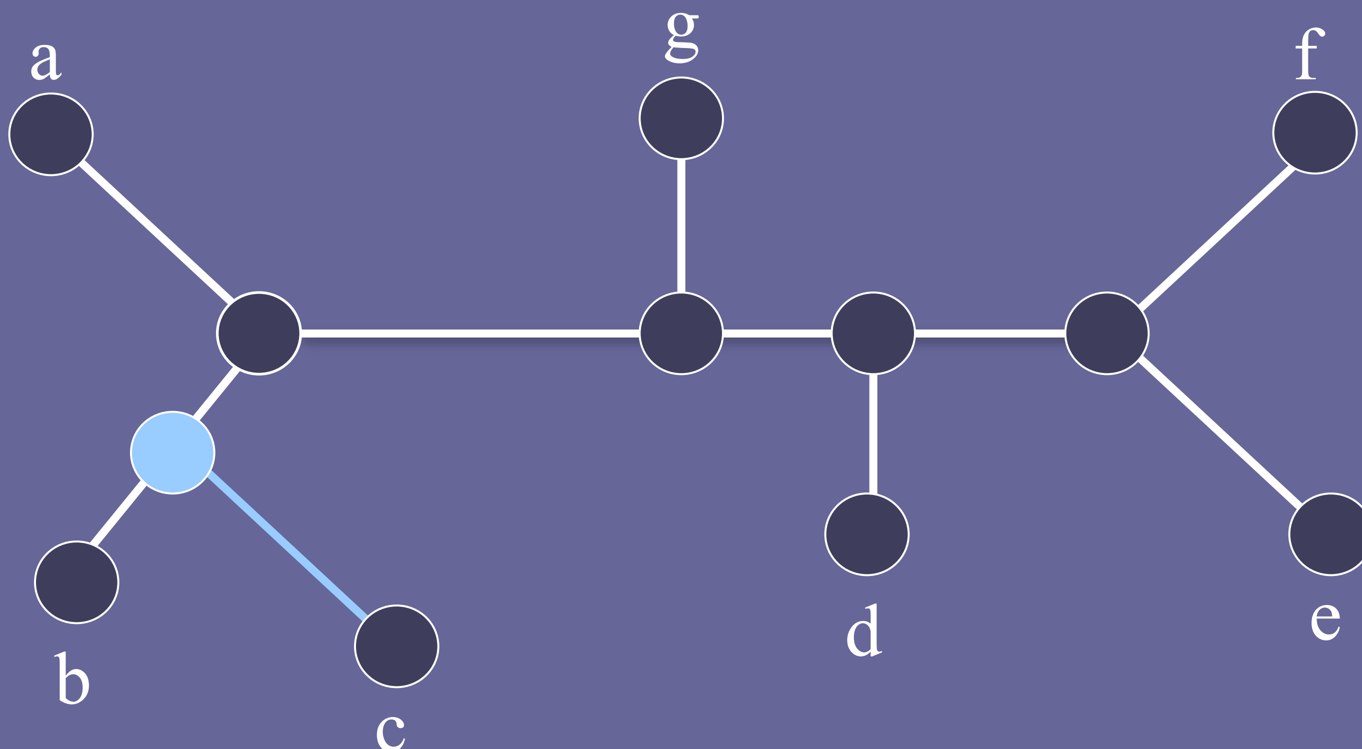
# SPR operation



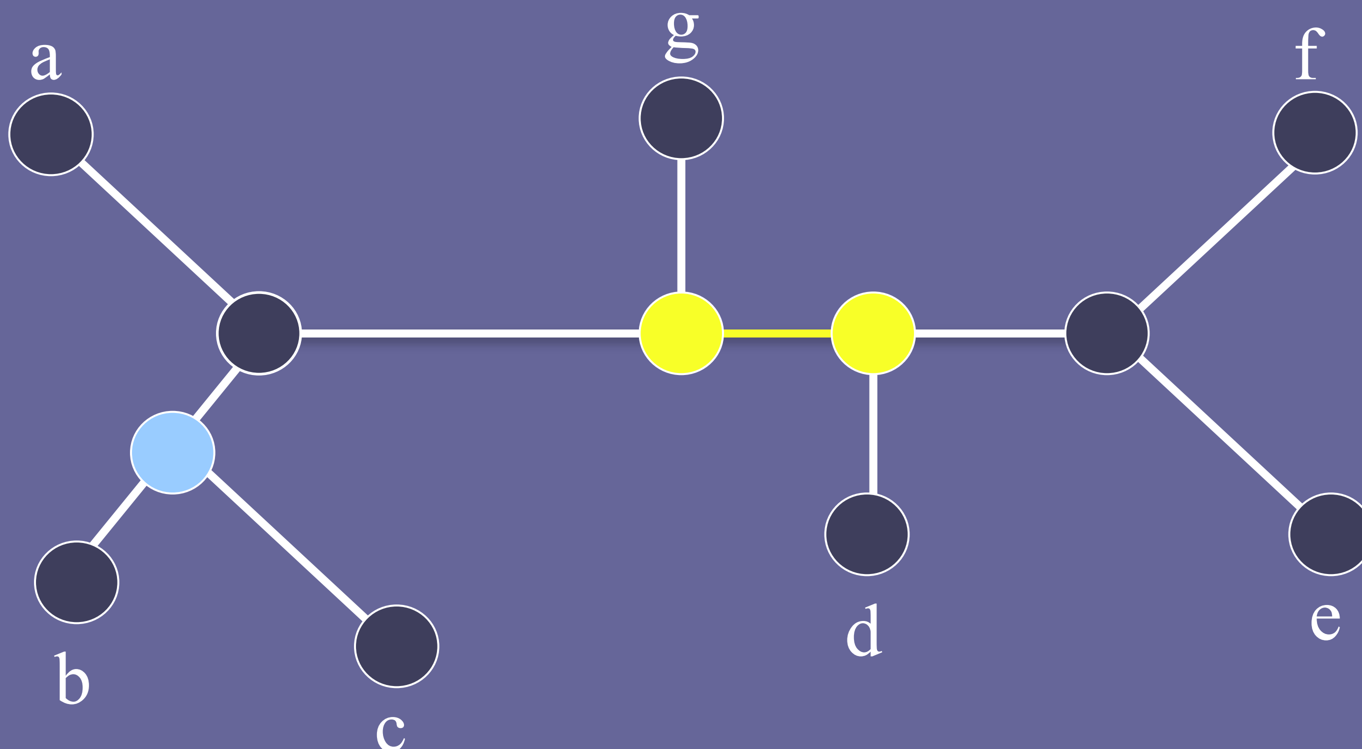
# SPR operation



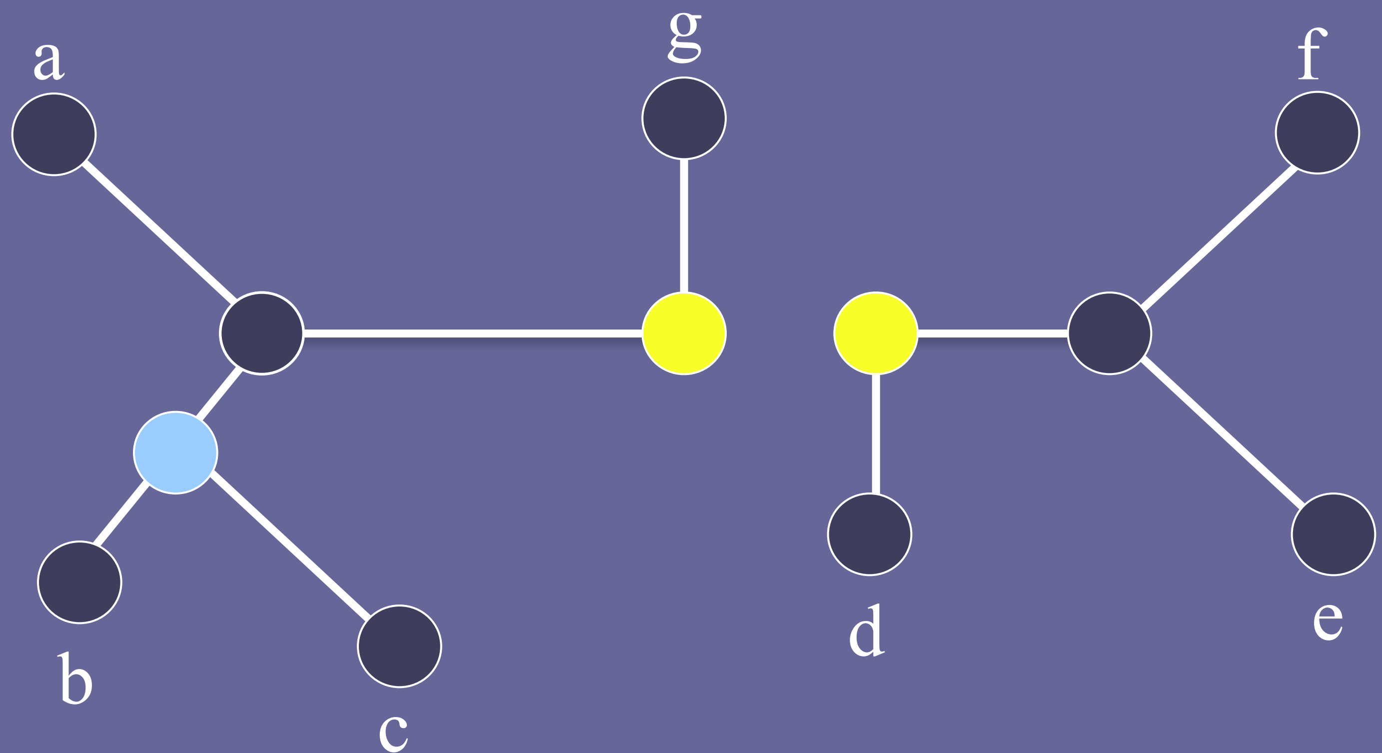
# SPR operation



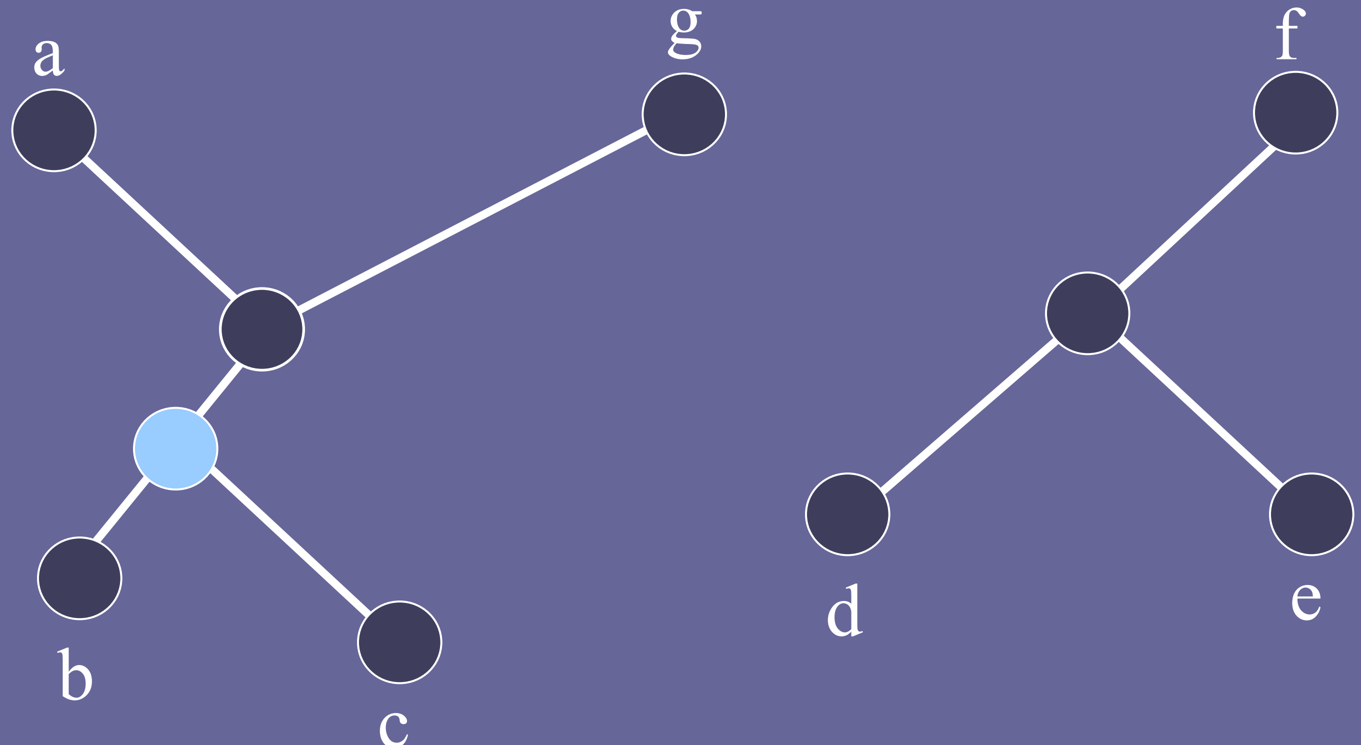
# TBR operation



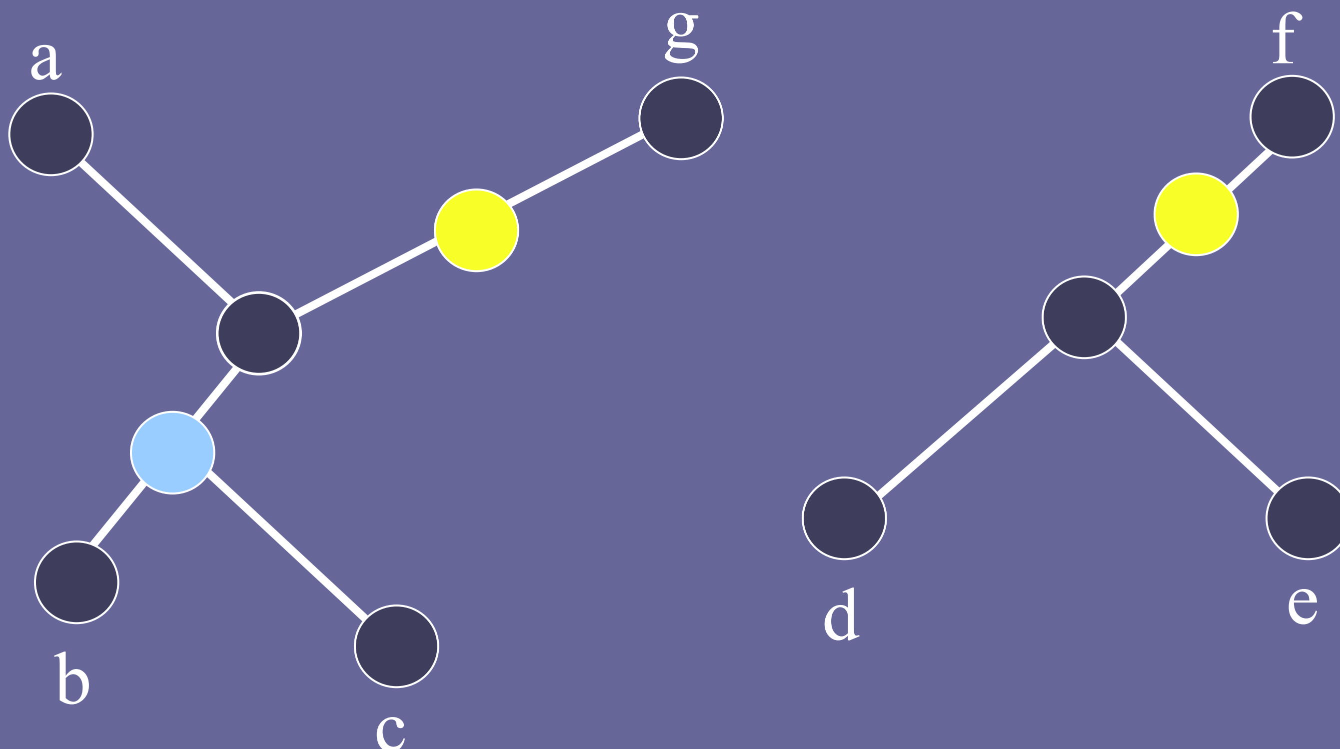
# TBR operation



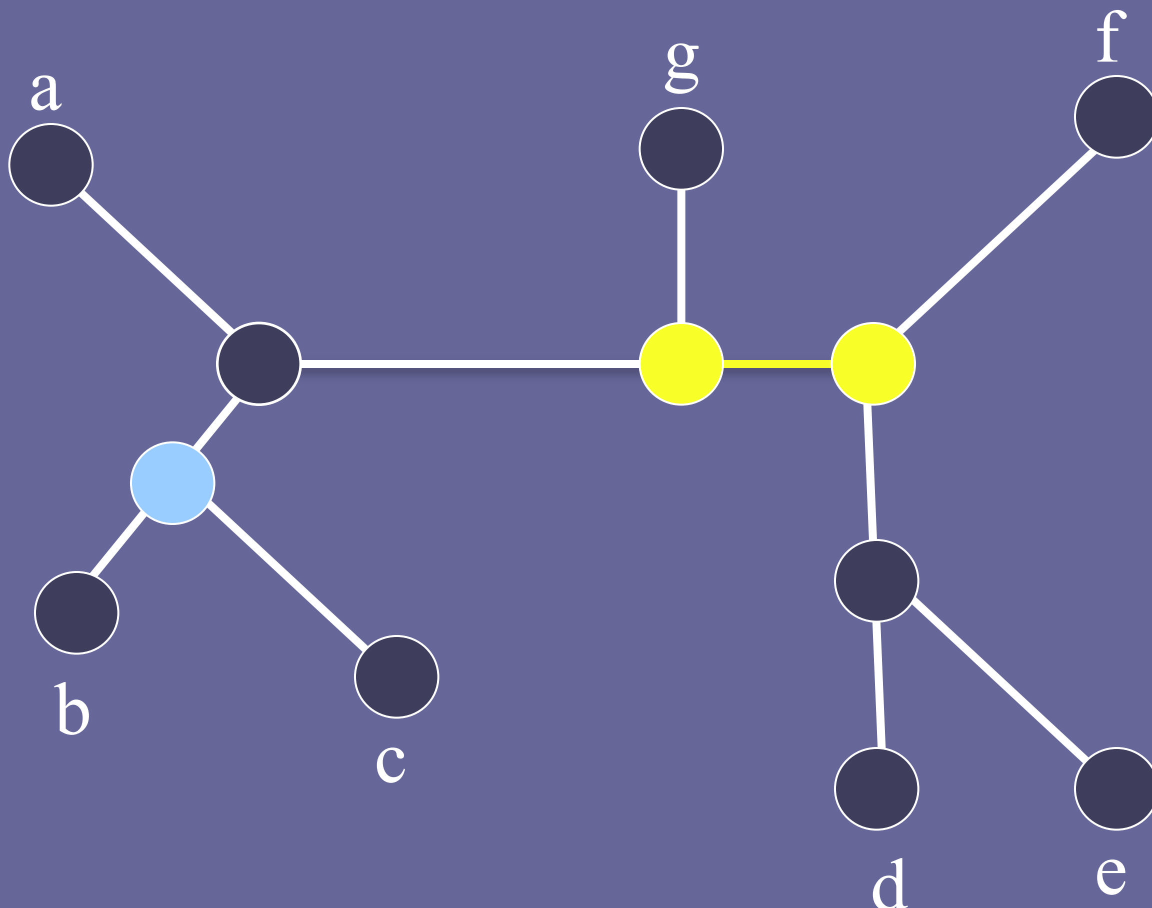
# TBR operation



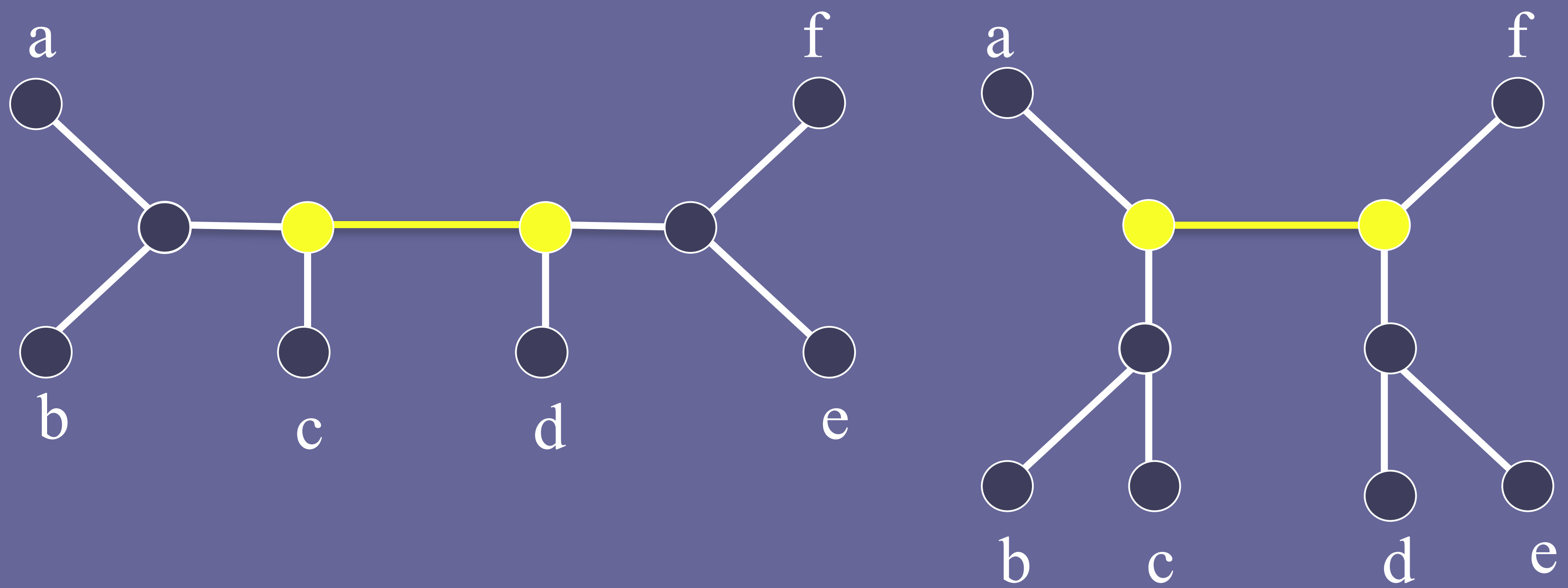
# TBR operation



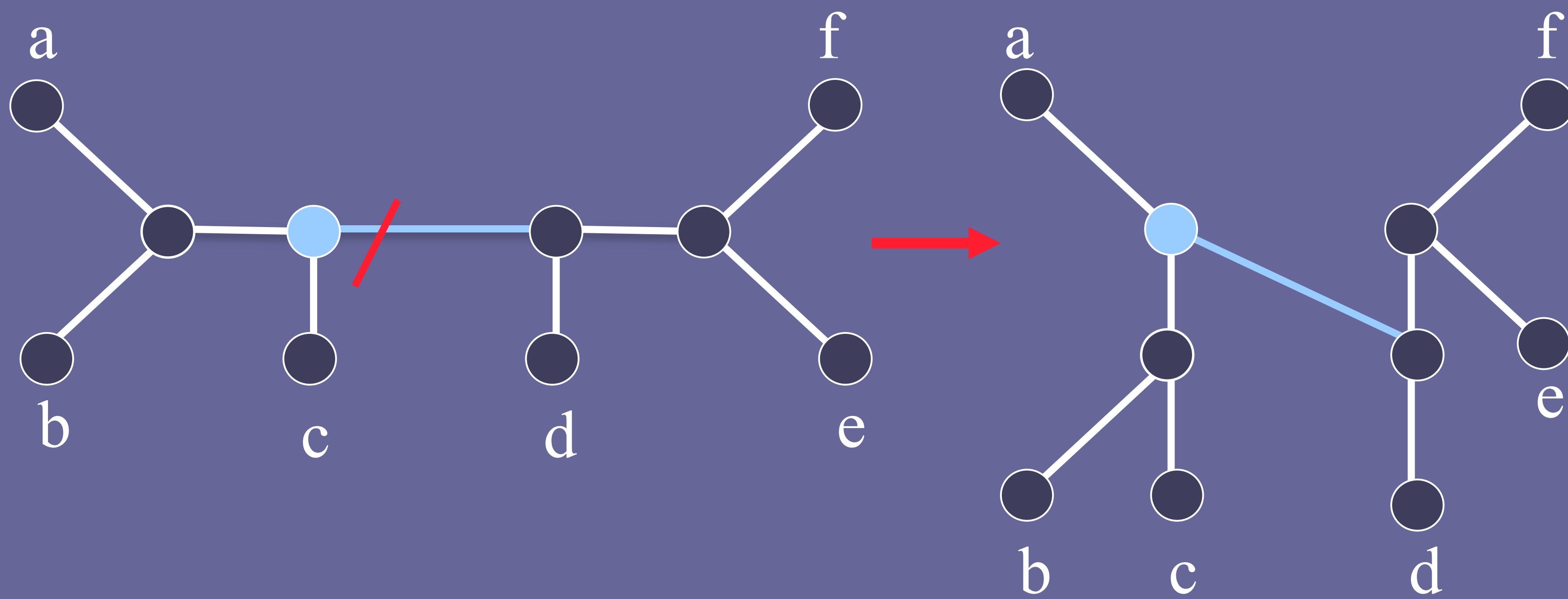
# TBR operation



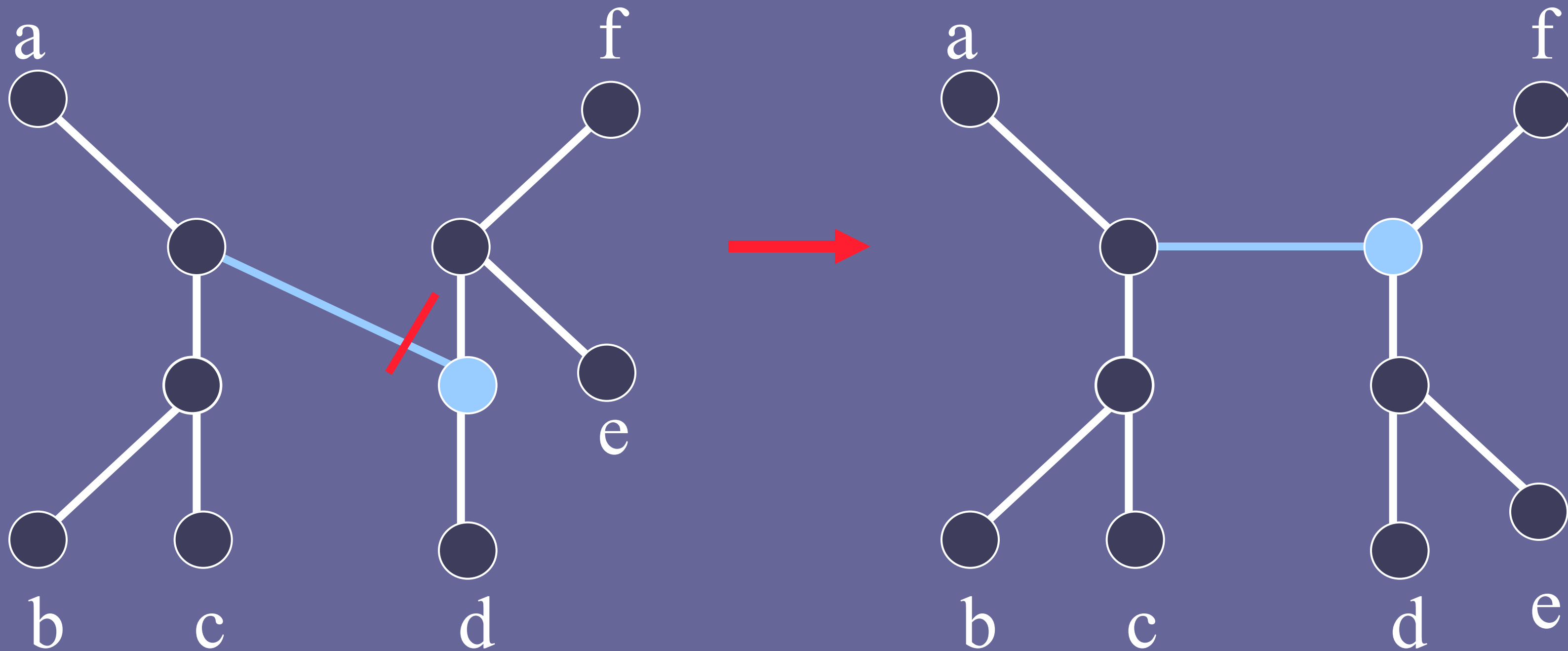
# 1 TBR = 1 or 2 SPR's



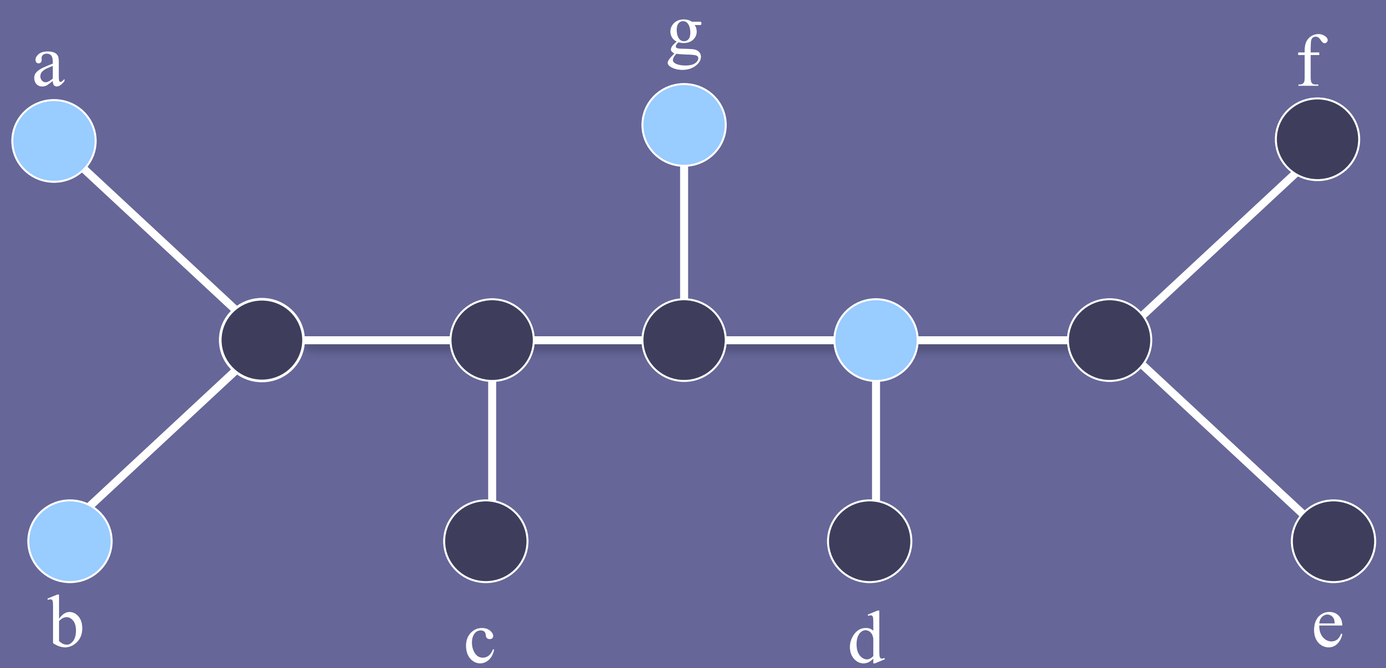
# 1 TBR = 1 or 2 SPR's



# 1 TBR = 1 or 2 SPR's

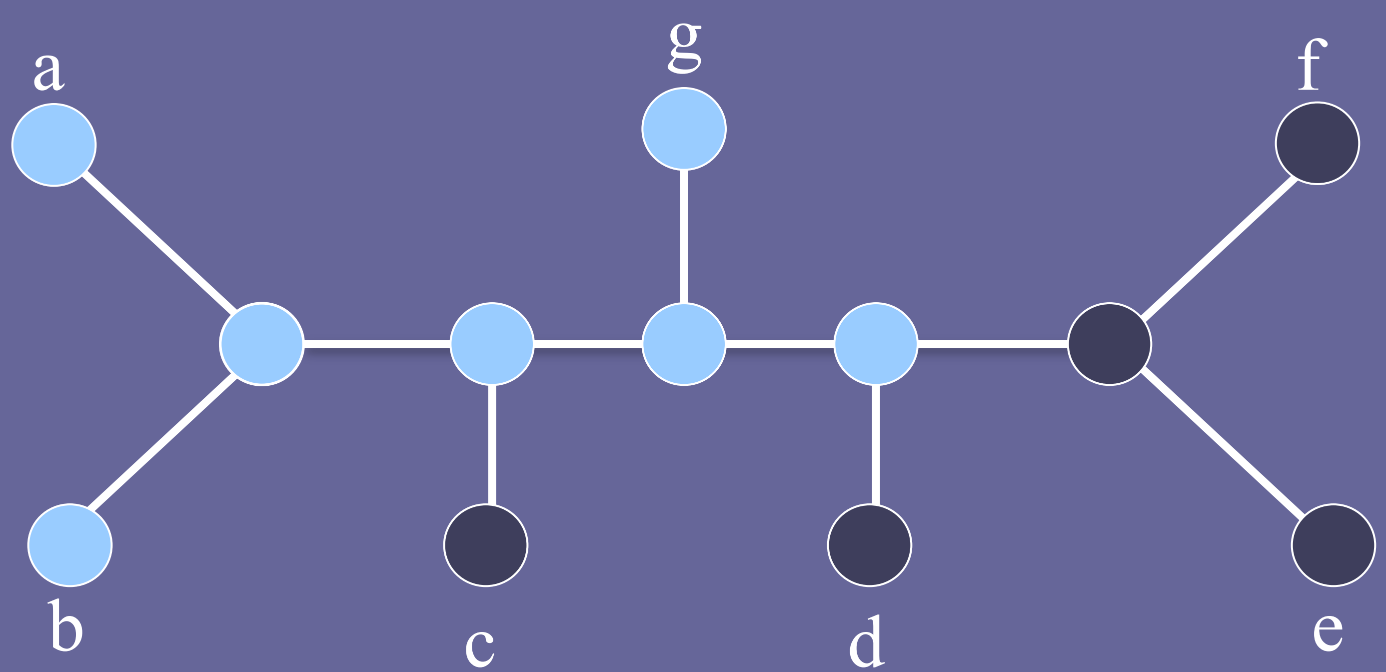


U = subset of V(T)

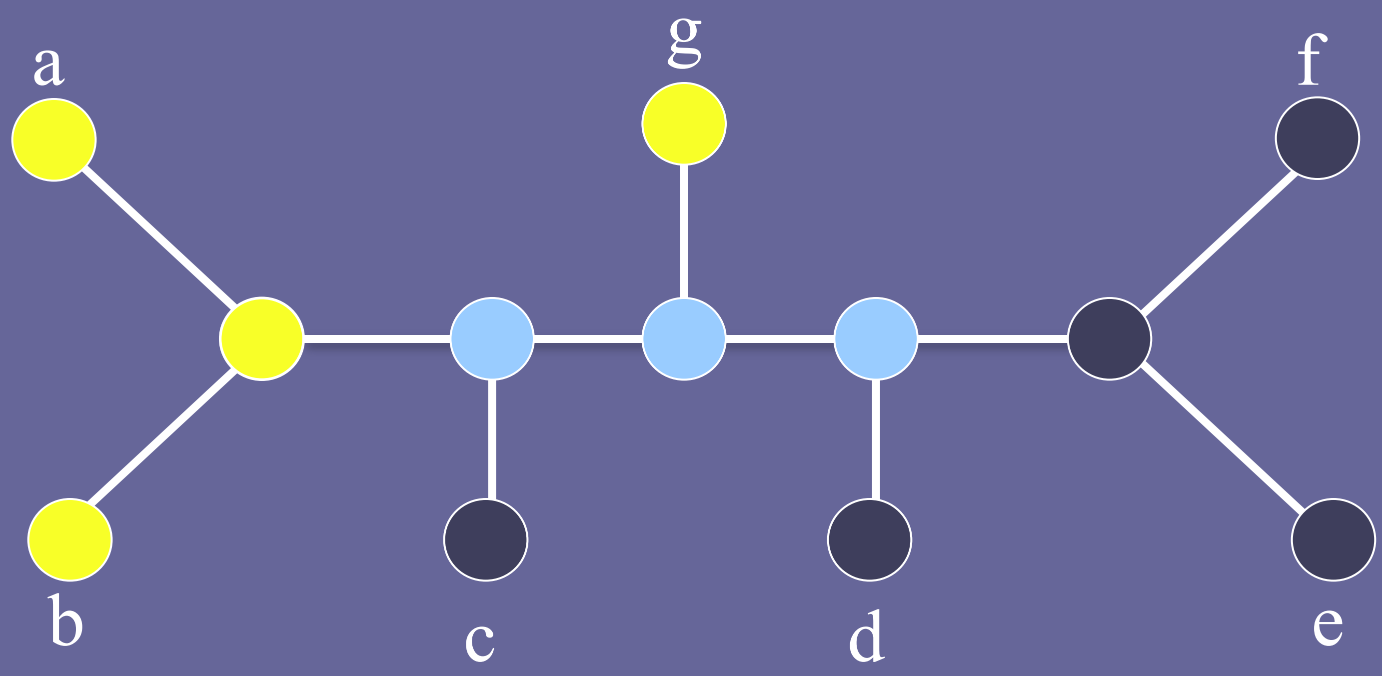


U = subset of V(T)

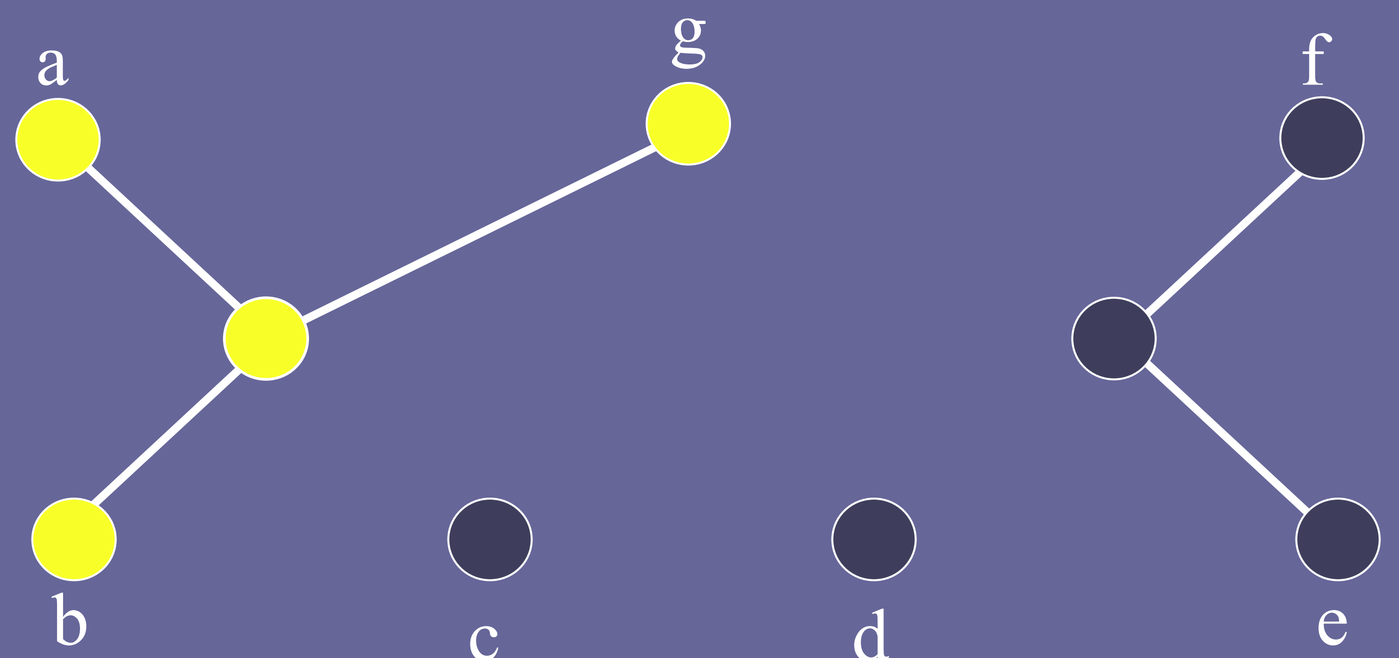
T[U] = minimal subtree connecting U in T.



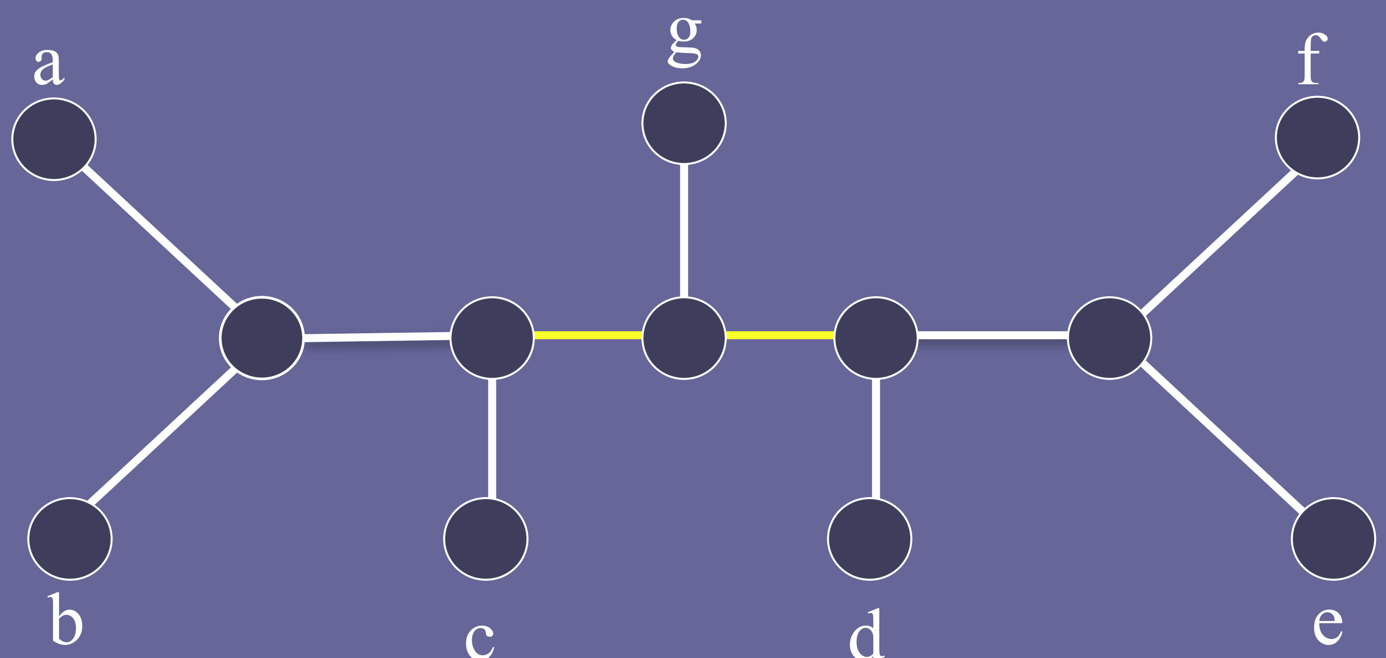
$\sigma(T[U])$ = phylogenetic tree obtained from T[U] by recursively contracting all degree 2 vertices and recursively removing any non-leaf degree 1 vertex



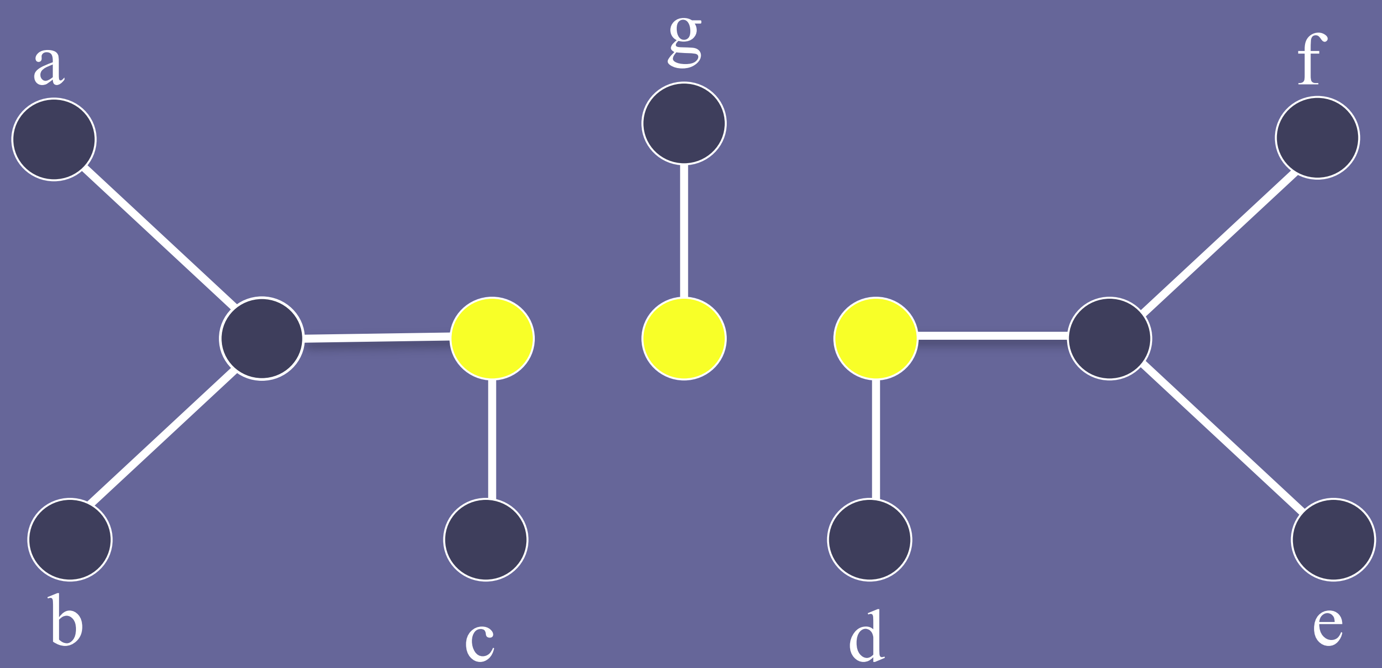
σ(T[U]) = phylogenetic tree obtained from T[U] by recursively contracting all degree 2 vertices and recursively removing any non-leaf degree 1 vertex



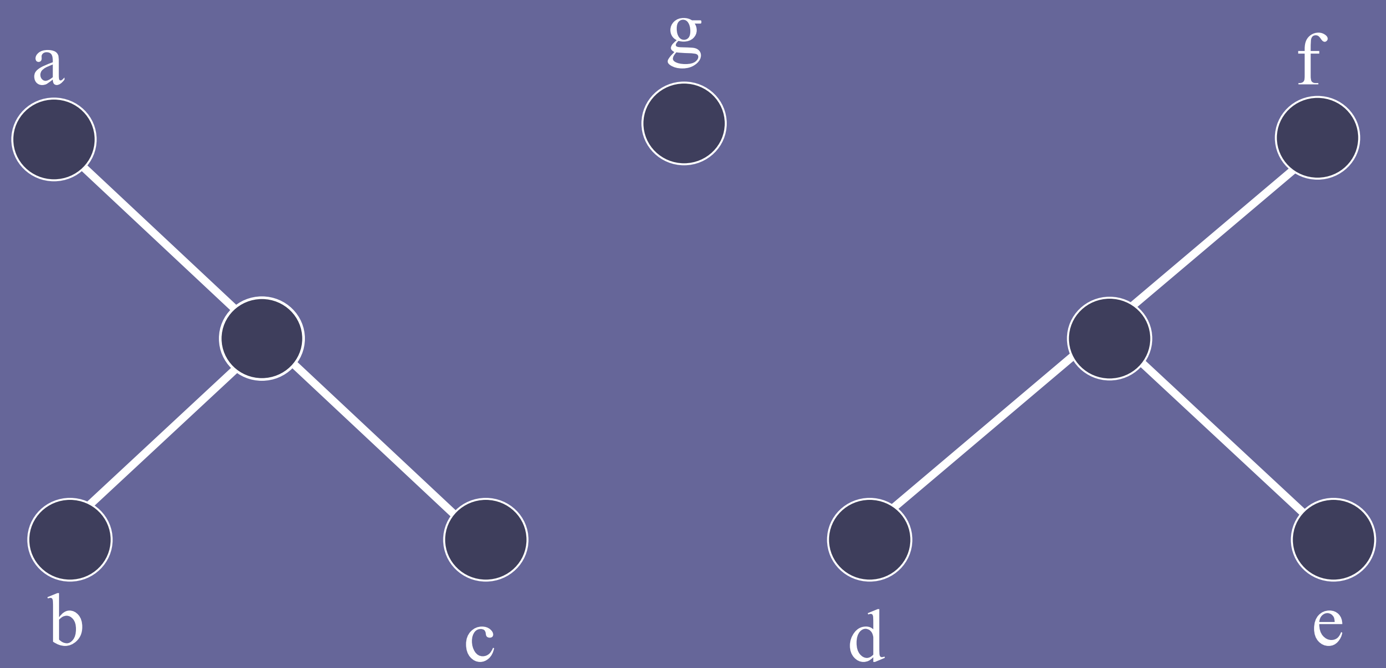
E = subset of E(T)



$\tau(T \setminus E)$ = phylogenetic forest obtained from $T \setminus E$, by recursively contracting all degree 2 vertices and recursively removing any non-leaf degree 1 vertex



$\tau(T \setminus E)$ = phylogenetic forest obtained from $T \setminus E$, by recursively contracting all degree 2 vertices and recursively removing any non-leaf degree 1 vertex



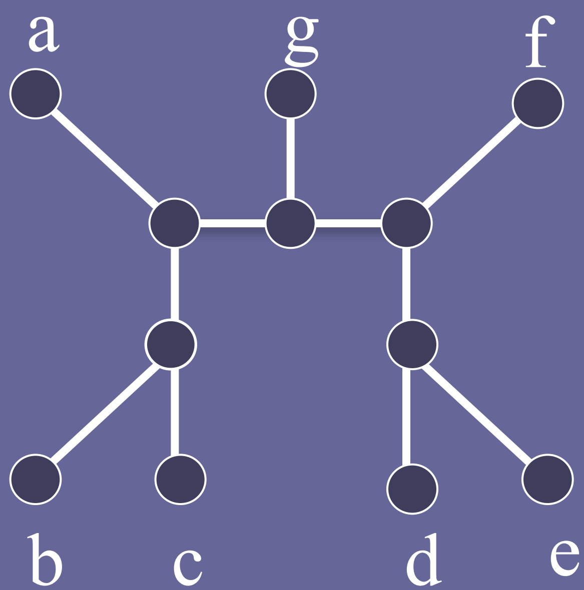
# Agreement forest for two X-trees $T_1$ and $T_2$

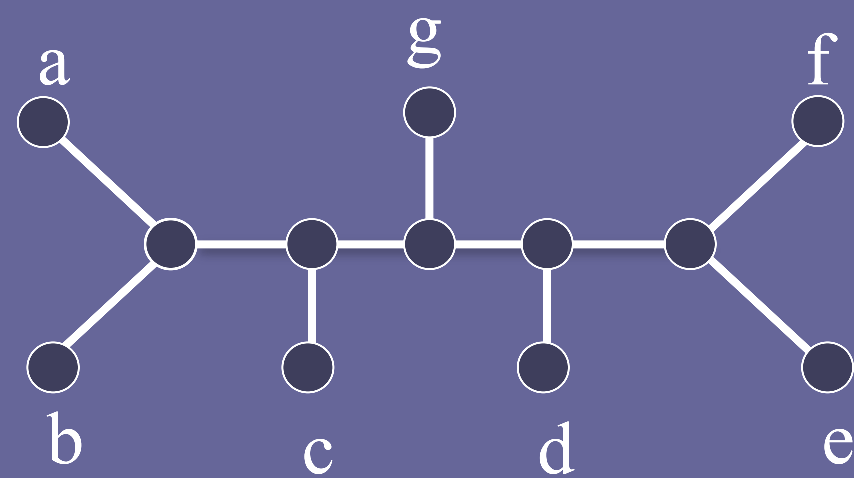
$\mathcal{F} = \{t_1, t_2, \ldots t_k\}$ a collection of phylogenetic trees such that

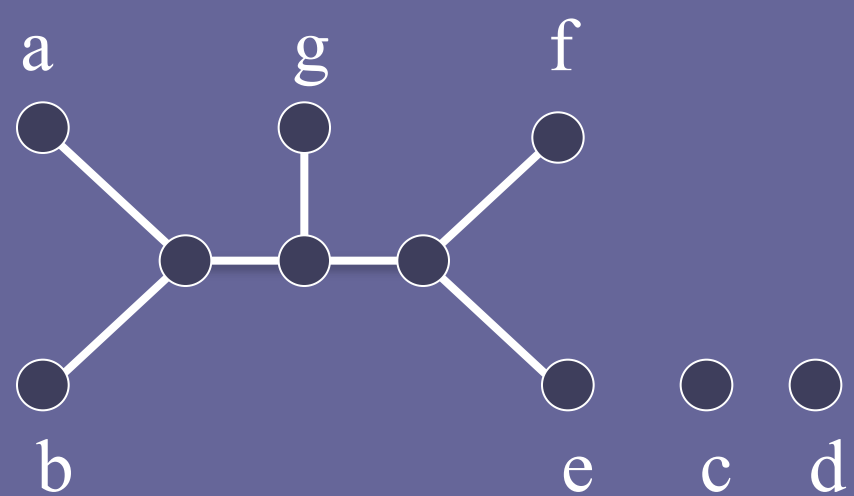
1. $\mathcal{L}(t_1), \ldots, \mathcal{L}(t_k)$ partitions X

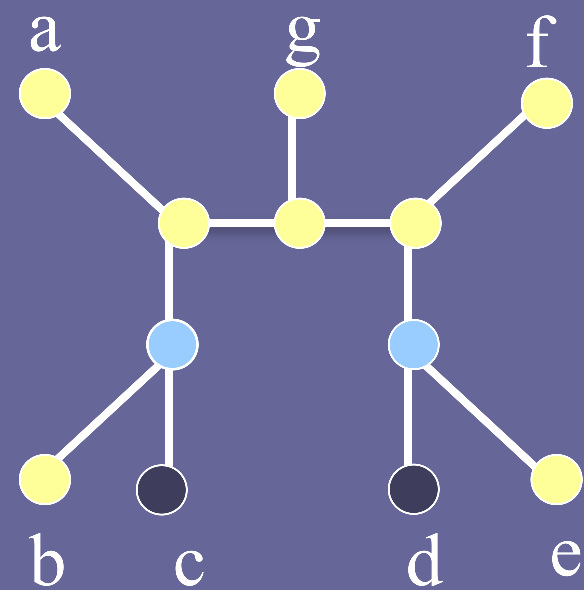
2. $\forall j \in \{1,\ldots, k\} \quad t_j = \sigma(T_1[L(t_j)]) = \sigma(T_2[L(t_j)])$

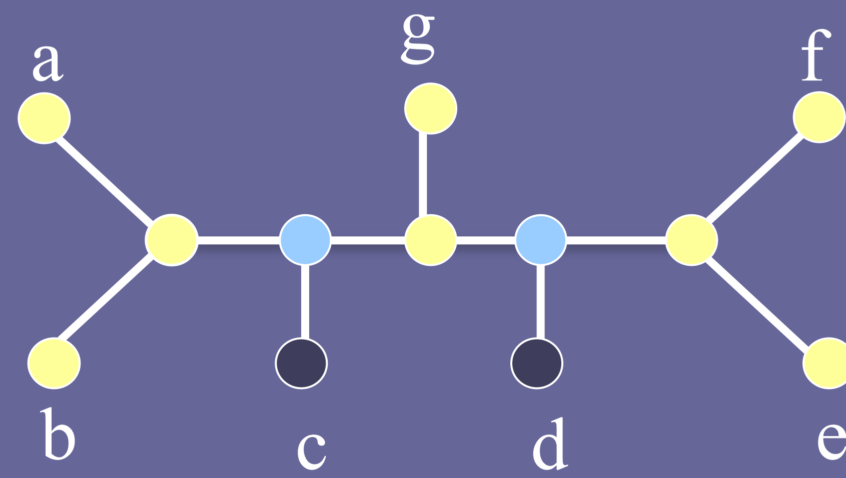
3. for i = 1, 2 the trees $\{T_i[L(t_j)] : 1 \leq j \leq k\}$ are vertex disjoint subtrees of $T_i$

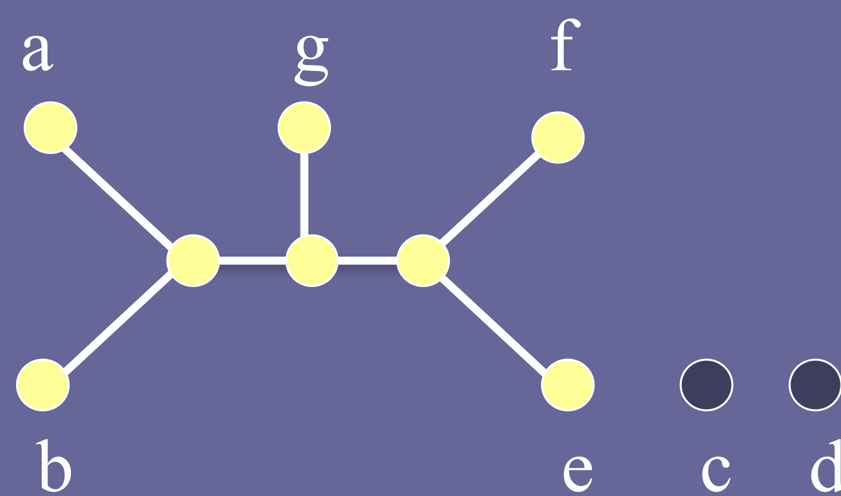

a
g
f
b
c
d
e

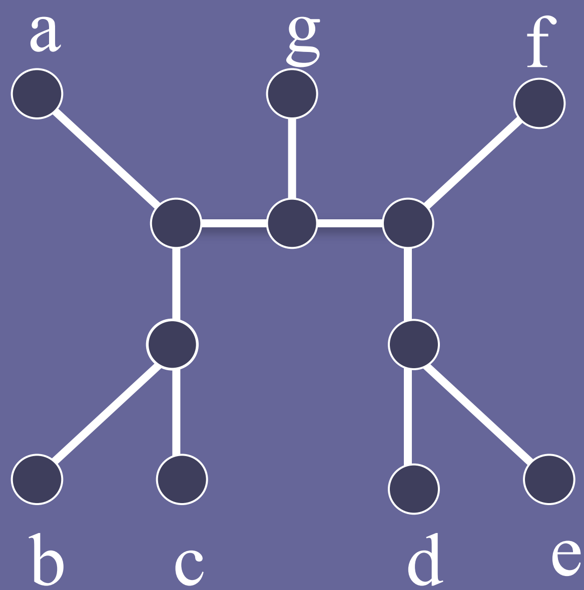

a
g
f
b
c
d
e

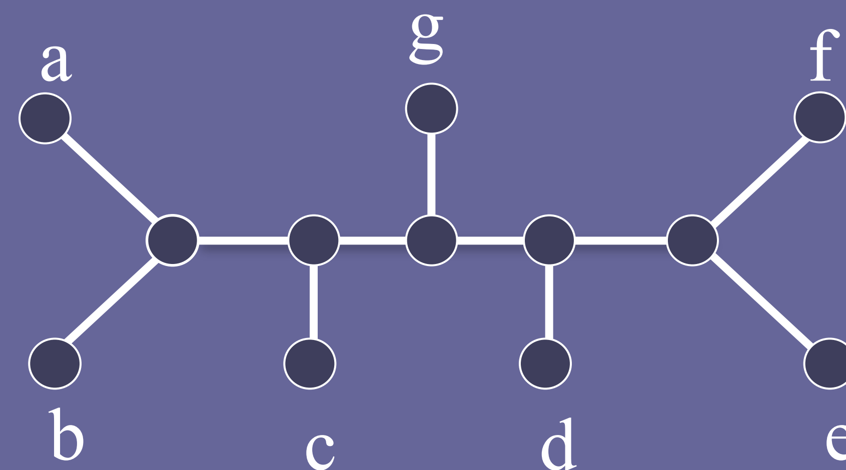

a
g
f
b
e
c
d

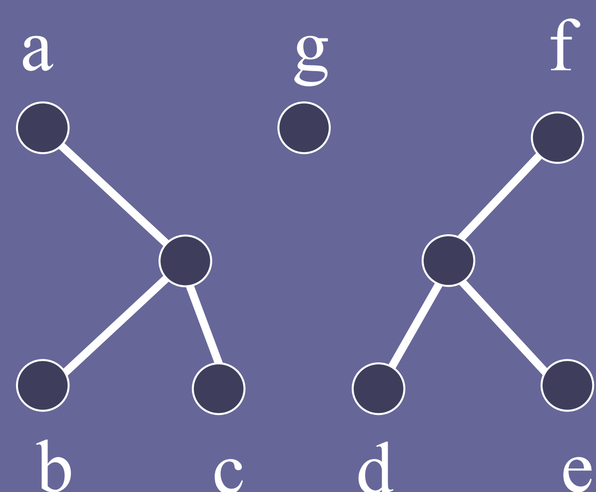

a
g
f
b
c
d
e

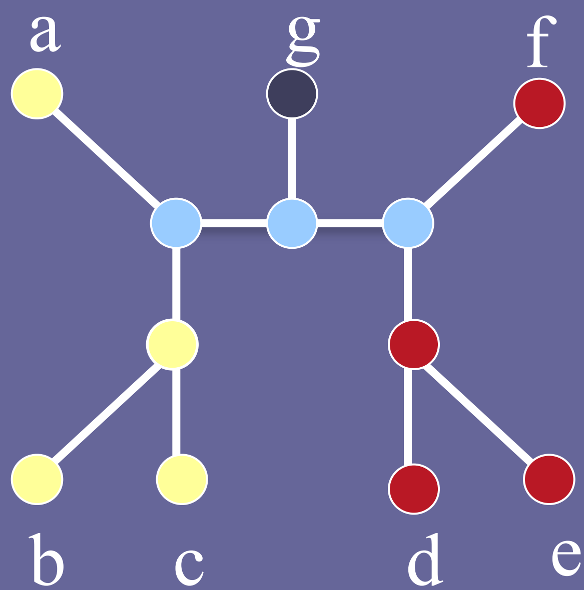

a
g
f
b
c
d
e

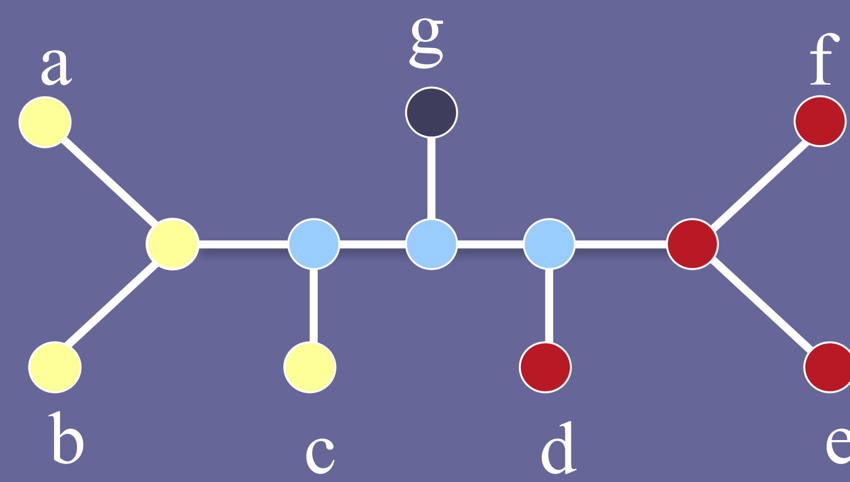

a
g
f
b
e
c
d

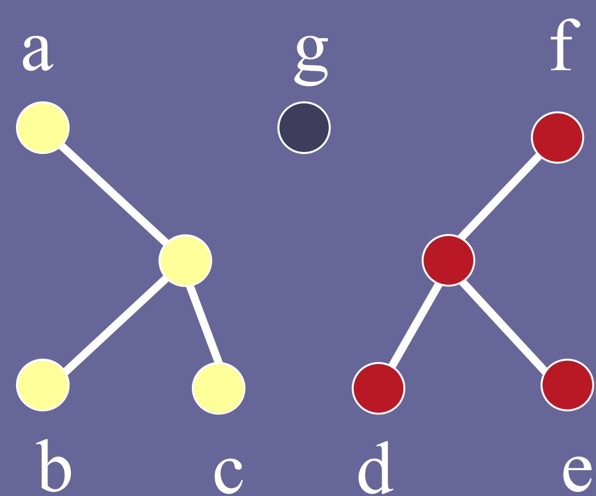

a
g
f
b
c
d
e


a
g
f
b
c
d
e


a
g
f
b
c
d
e


a
g
f
b
c
d
e


a
g
f
b
c
d
e


a
g
f
b
c
d
e

# Maximum agreement forest (MAF) for $T_1$ and $T_2$

an agreement forest with fewest components possible

$F$ an agreement forest for $T_1$ and $T_2$ with $|F| = k$ minimized

$k - 1 = m(T_1,T_2)$ is equal to the TBR distance for $T_1$ and $T_2$
$m(T_1,T_2)$ is at least half the SPR distance for $T_1$ and $T_2$

computing $m(T_1,T_2)$ is NP-hard
(Hein et al. 1996, Allen and Steel 2000)

# Parameterized MAF problem

k-Maximum Agreement Forest Problem:

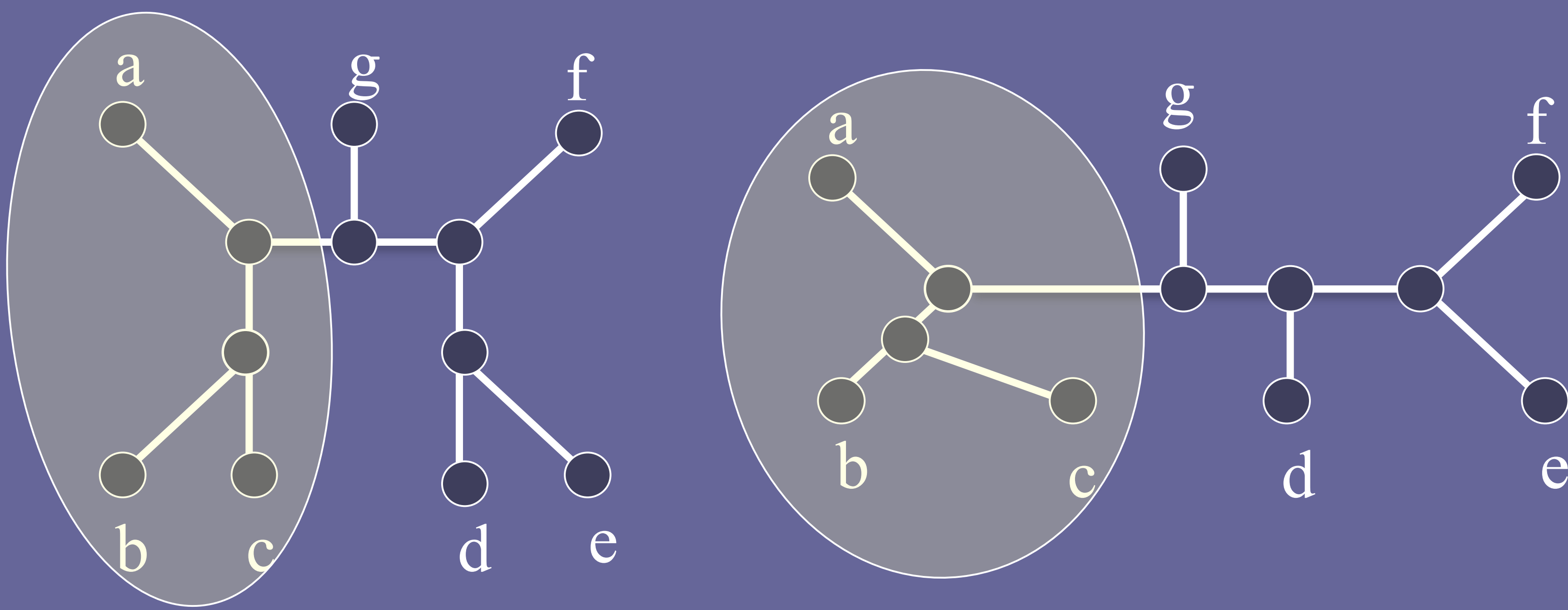
Input: pair of phylogenetic X-trees $T_1$ and $T_2$
Parameter: k
Output: agreement forest $F = \{t_1, t_2, \ldots t_{k'}\}$ for $T_1$ and $T_2$
with $k' \leq k+1$, or 'NO' if $m(T_1,T_2) > k$

# FPT algorithm for k-MAF problem

Kernelization:

Rule 1:

contract identical pendant subtrees into single leaves with new label

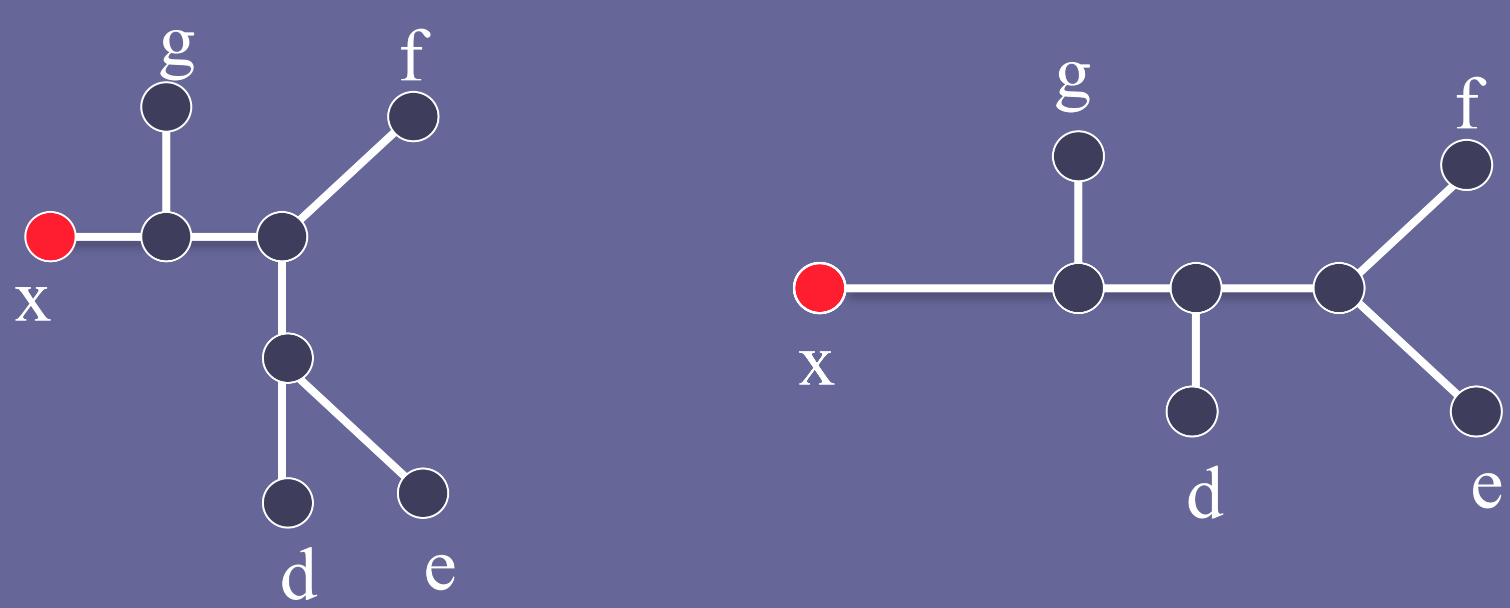


# FPT algorithm for k-MAF problem

Kernelization:

Rule 1:

contract identical pendant subtrees into single leaves with new label

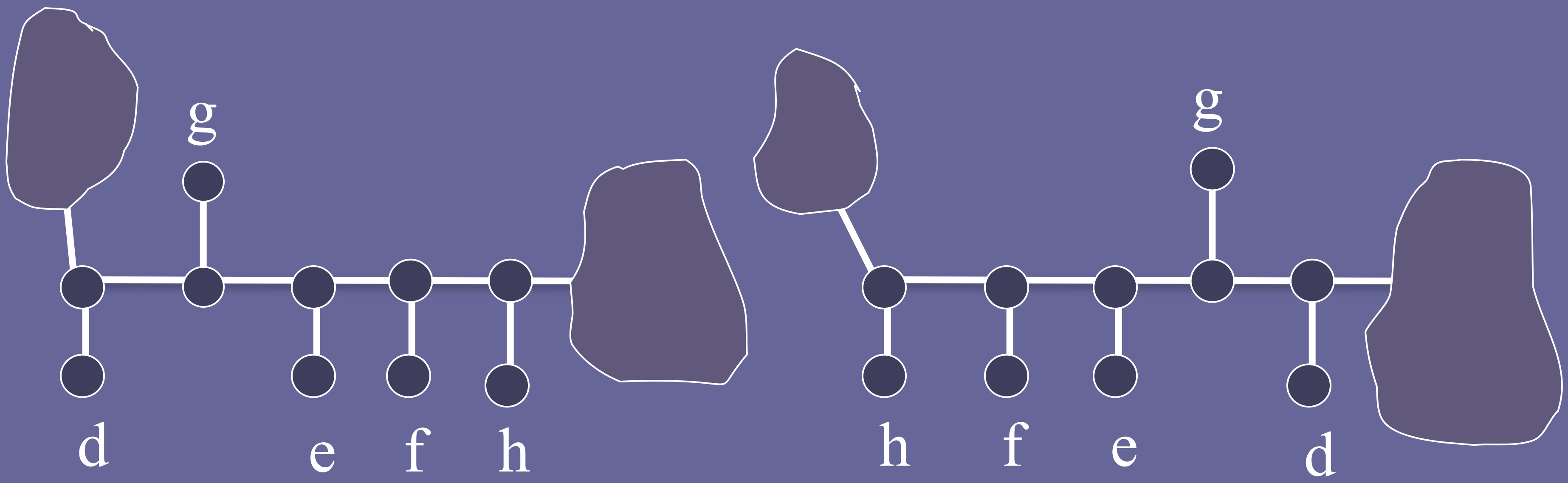


# FPT algorithm for k-MAF problem

Kernelization:

Rule 2:

contract identical chains of pendant subtrees into 3-leaf chains with labels preserving orientation

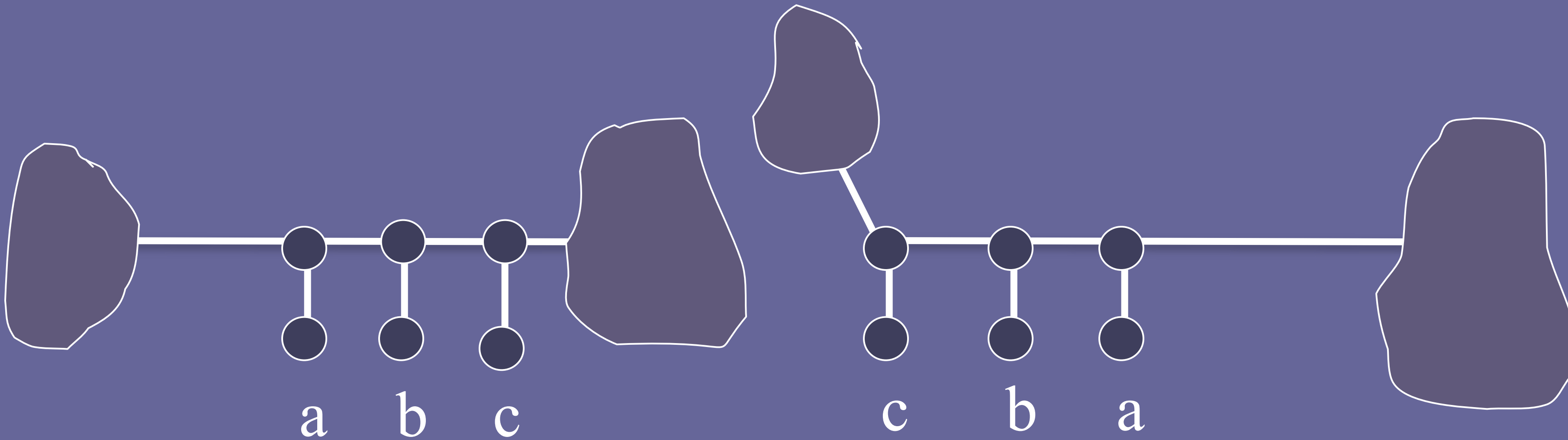


# FPT algorithm for k-MAF problem

Kernelization:

Rule 2:

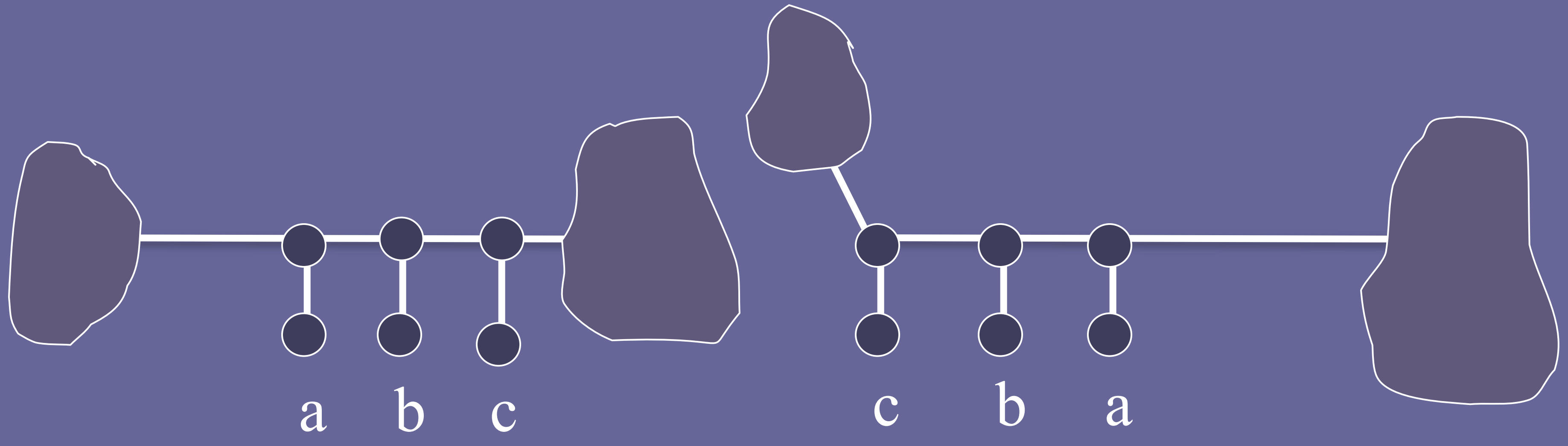
contract identical chains of pendant subtrees into 3-leaf chains with labels preserving orientation



# FPT algorithm for k-MAF problem

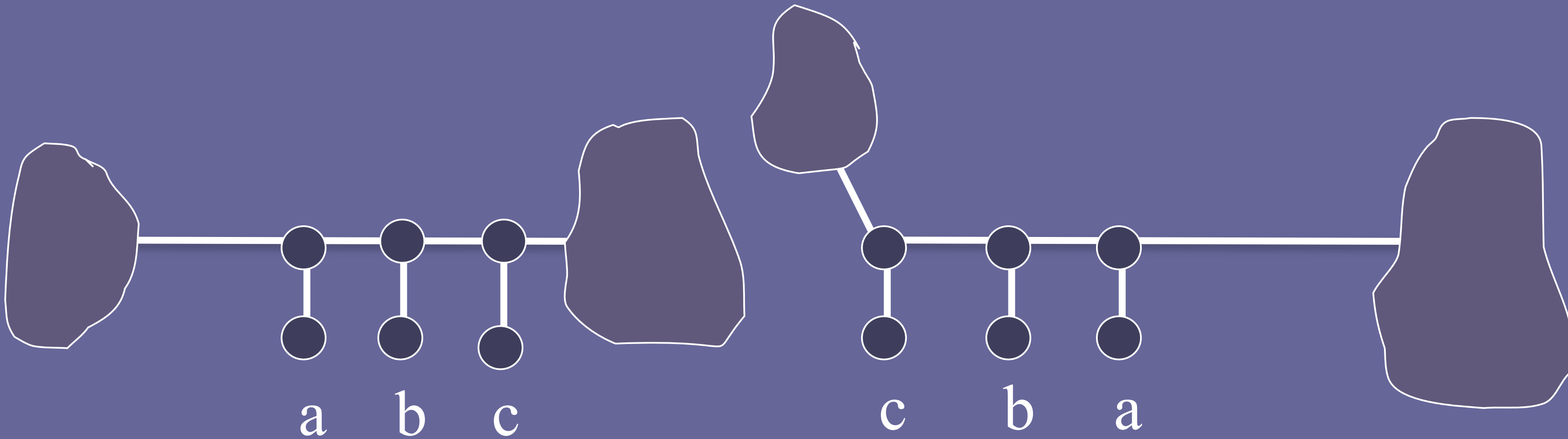
Kernelization:

Reduced trees are either a NO instance
or the number of leaves in each tree is bounded by 28k.



# FPT algorithm for k-MAF problem

Kernelization:
If > 28k leaves in reduced trees then answer NO
else brute force search all k-subsets of 56k edges in $T_1$
Total time = $O(56k^k) + (|X|^c)$



# FPT algorithm for k-MAF problem

Search tree strategy:

Phase 1: look for *minimal incompatible quartets* in the current forest wrt $T_2$

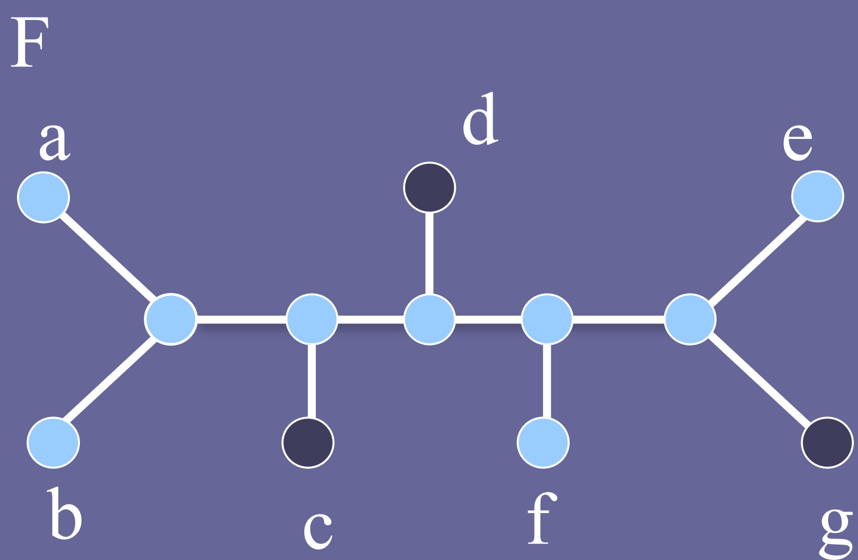
remove such a quartet by ‘depronging’ the structure in exactly four ways, leading to four branches in our search tree.

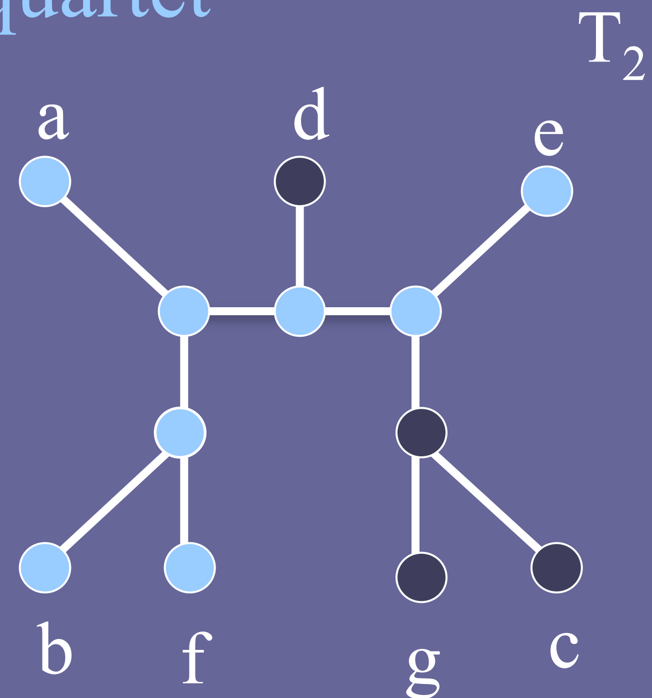
Phase 2: look for obstructions in the current forest wrt $T_2$

remove each obstruction in exactly two ways, leading to two branches in our search tree.

Running time $O(4^k \cdot k^5) + p(|X|)$ (when combined with kernelization.)

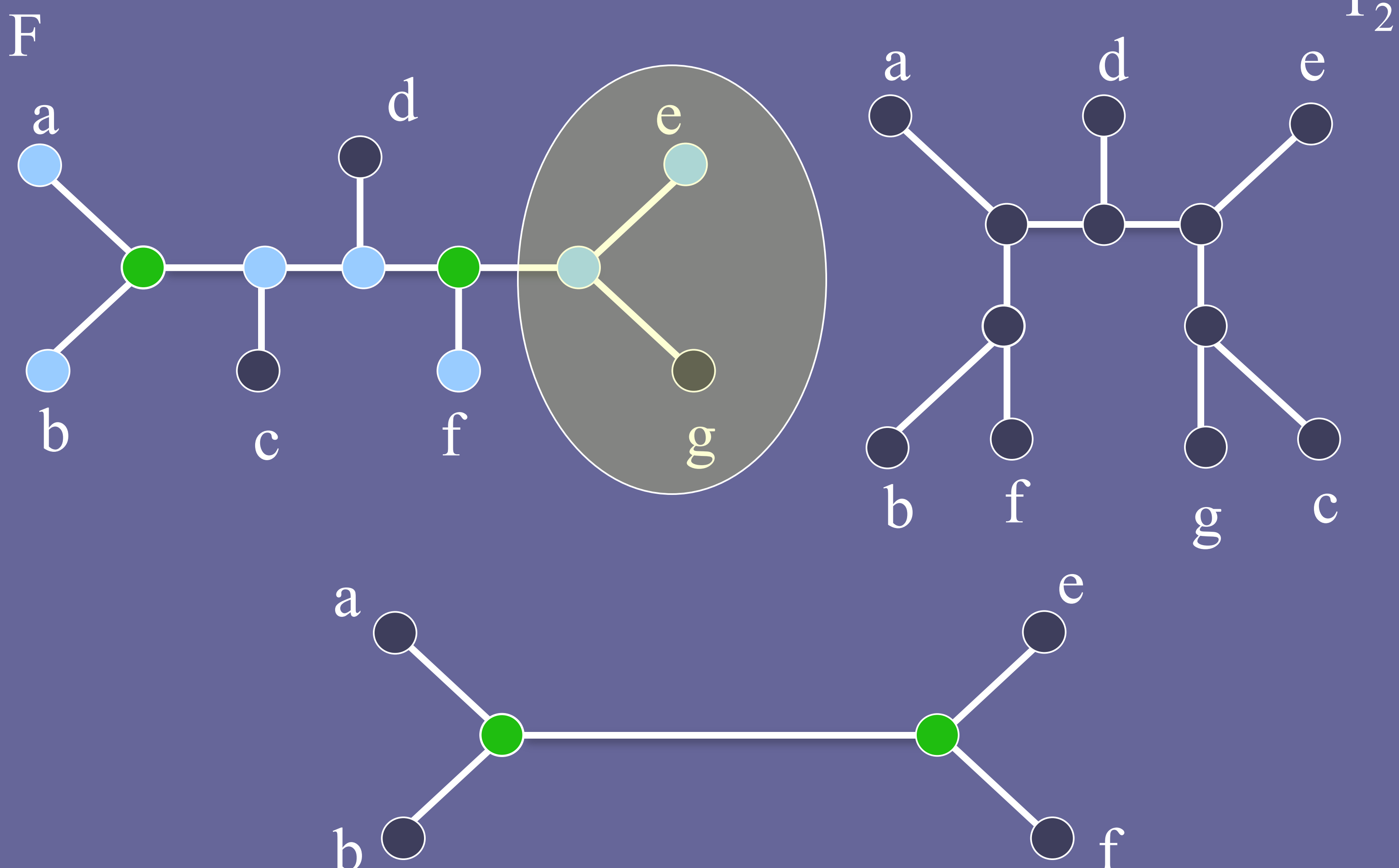
# Incompatible quartet



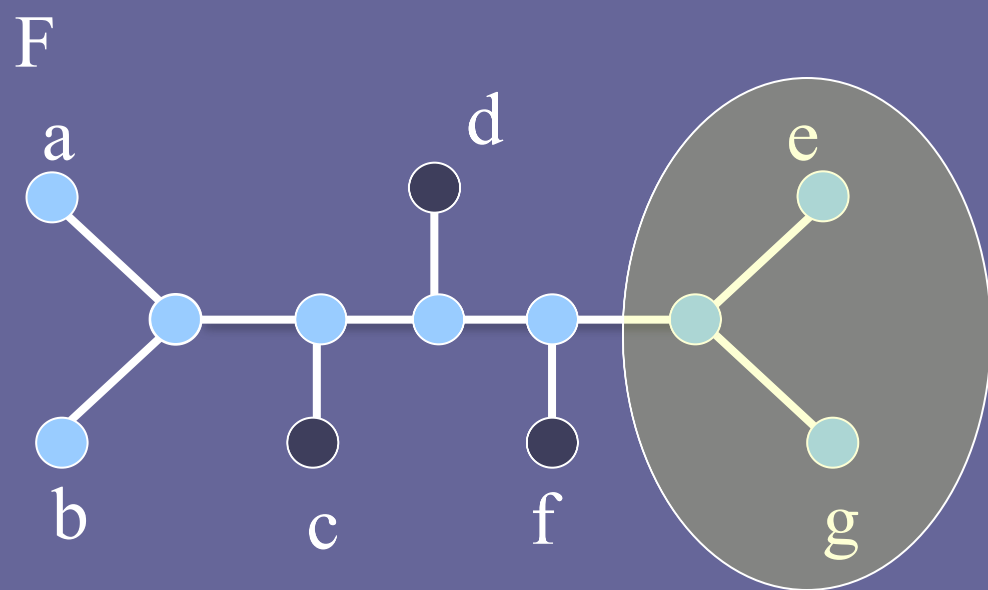



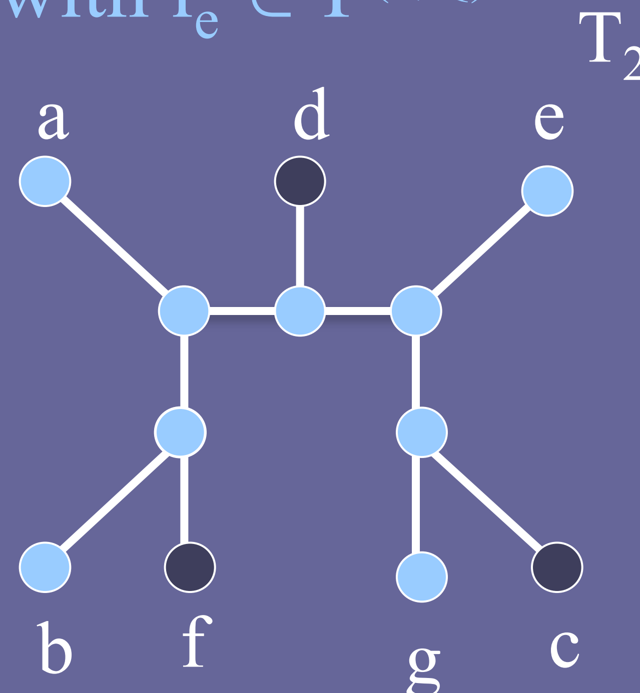

$Q = ab \mid ef$

# ‘e’ branch of Q in F, denoted by $F^{(e,Q)}$



# Replacing sibling of e with $l_e \in F^{(e,Q)}$

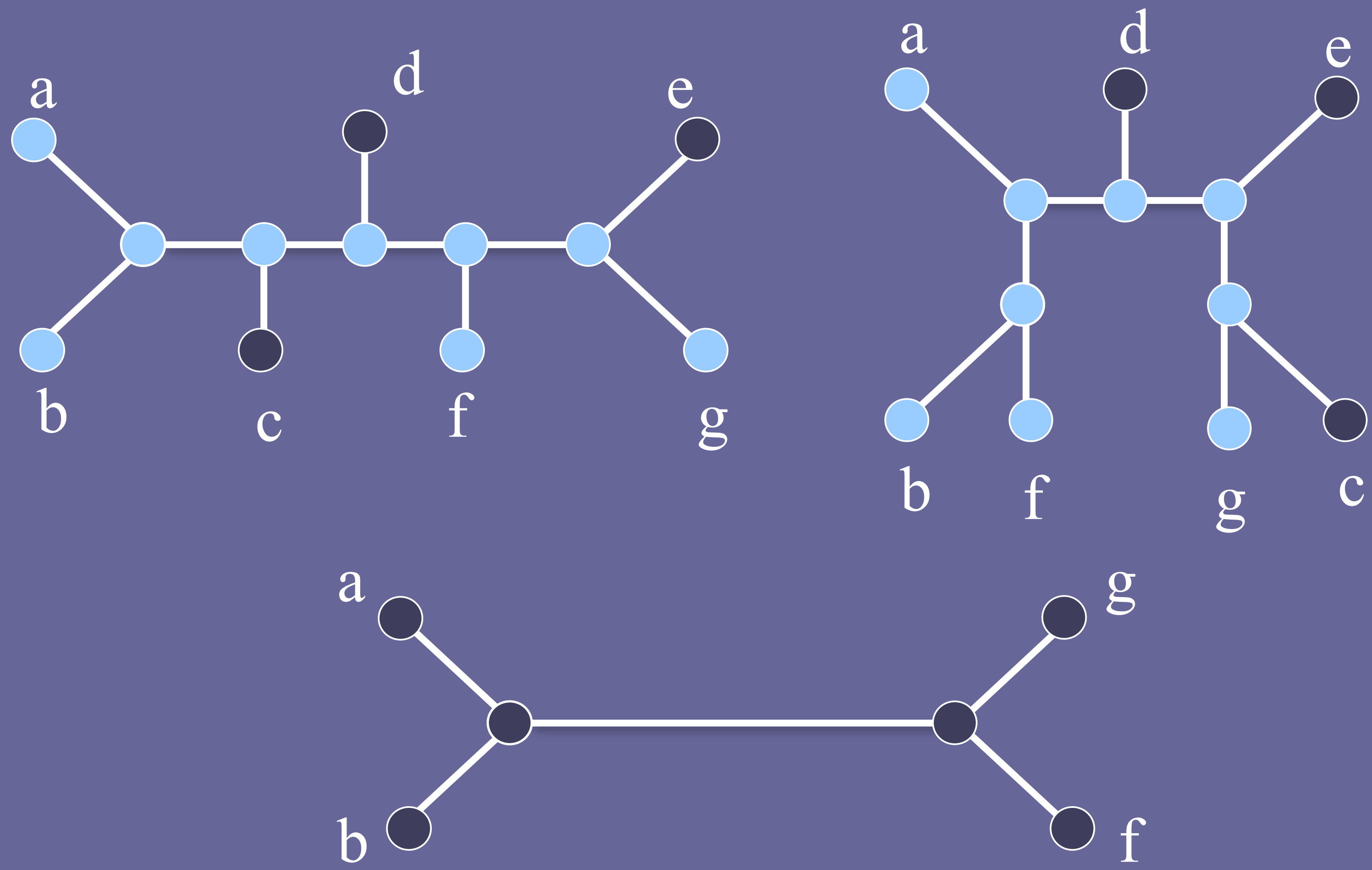




this quartet *is* compatible with $T_2$

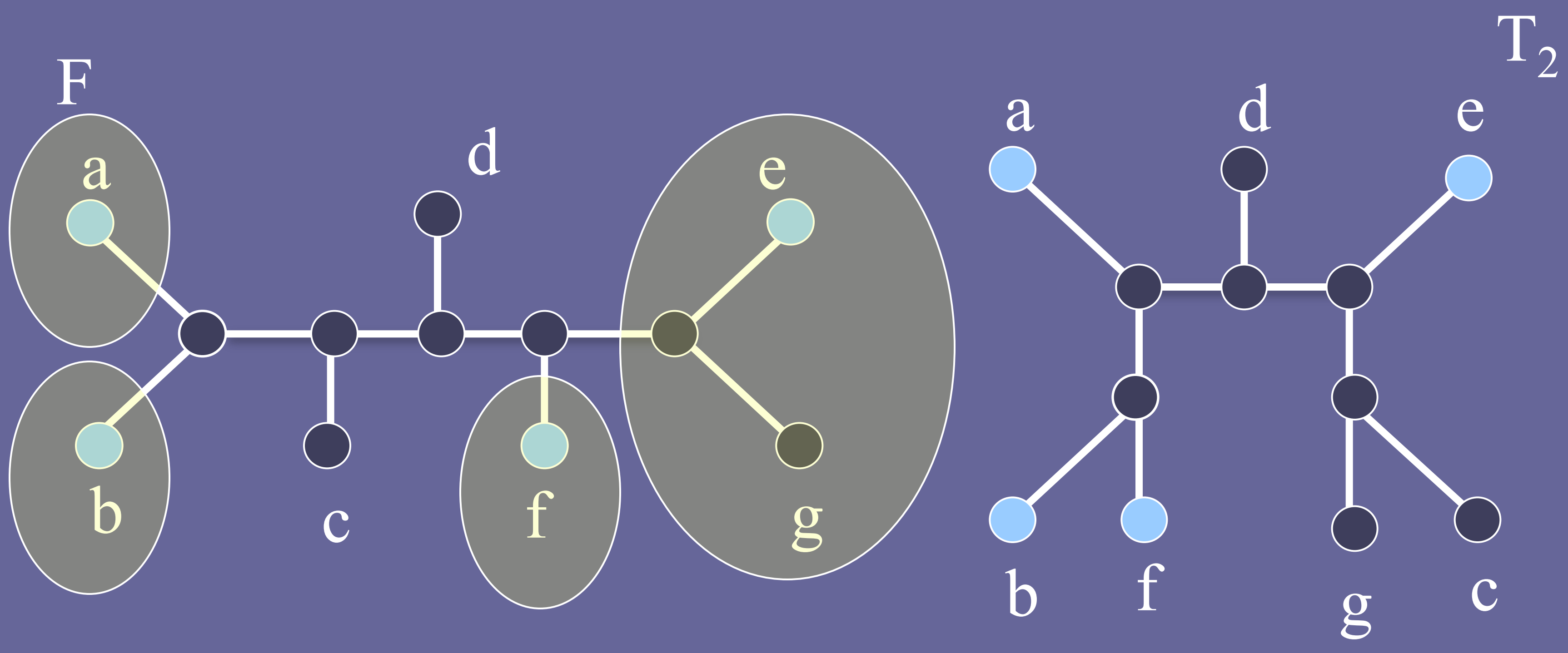
# Another incompatible quartet - ab | $l_e$f



# Minimal incompatible quartet

$Q = ab \mid ef$ is a *minimal incompatible quartet* in F wrt $T_2$ if

1. Q is *incompatible* with $T_2$
2. for all $x \in \{a, b, c, d\}$ and for all $l \in F^{(x,Q)}$, the quartet formed by replacing sibling of x with l is *compatible* with $T_2$



# Minimal incompatible quartet property

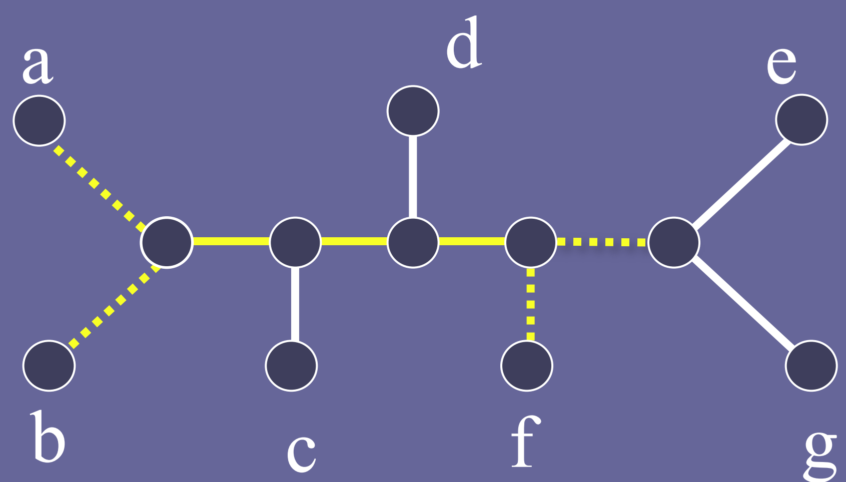
Let $Q = ab \mid ef$ be a minimal incompatible quartet in F wrt $T_2$
then, for *any* set of leaves $\{l_a, l_b, l_e, l_f\}$ with $l_x$ in the 'x' branch
of Q wrt $T_1$
we have $l_a\ l_b \mid l_e\ l_f$ an incompatible quartet in F wrt $T_2$.

Thus, for any MAF $F' = \{t_1, t_2, \ldots t_k\}$ for F and $T_2$,
no set of the form $\{l_a, l_b, l_e, l_f\} \subseteq t_i$, for any $1 \le i \le k$.

# Fork R induced by Q in F

Q = ab | ef


R

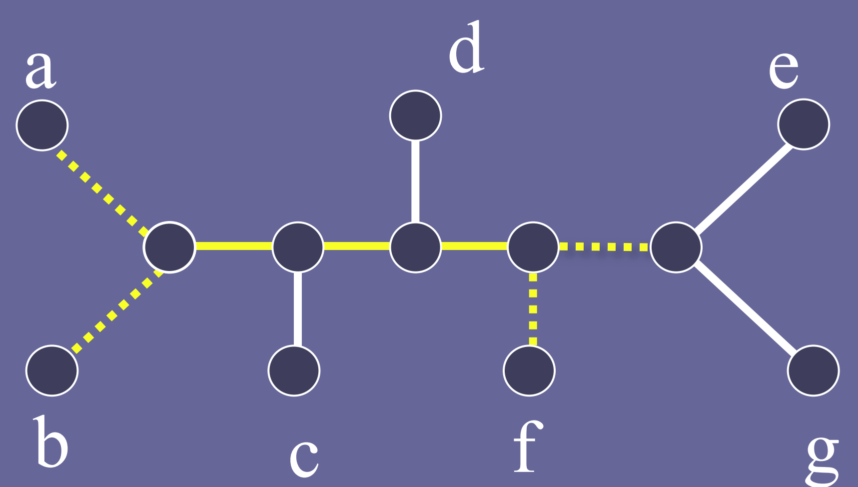



dashed edges are *prongs* of R

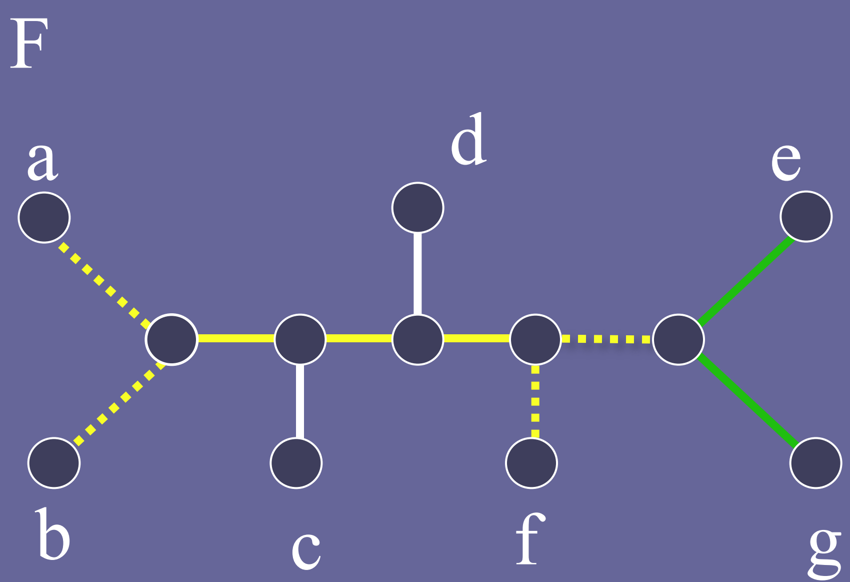
# Minimal incompatible quartet property

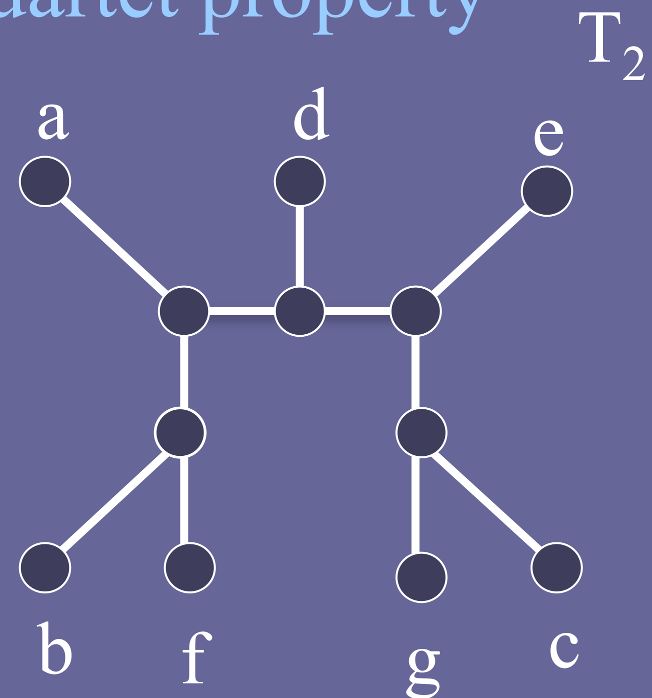
Q = ab | ef , a minimal incompatible quartet in F wrt $T_2$, with induced fork R let *F'* be an agreement forest for F and $T_2$.

any set of edges such that $\tau$(F \ E) produces *F'* from F will either contain one of the prongs of R, or an edge that can be exchanged for one of the prongs of R to produce E', such that $\tau$(F \ E') produces *F'* from F.



# Minimal incompatible quartet property




$Q = ab \mid ef$

# FPT algorithm: Phase I

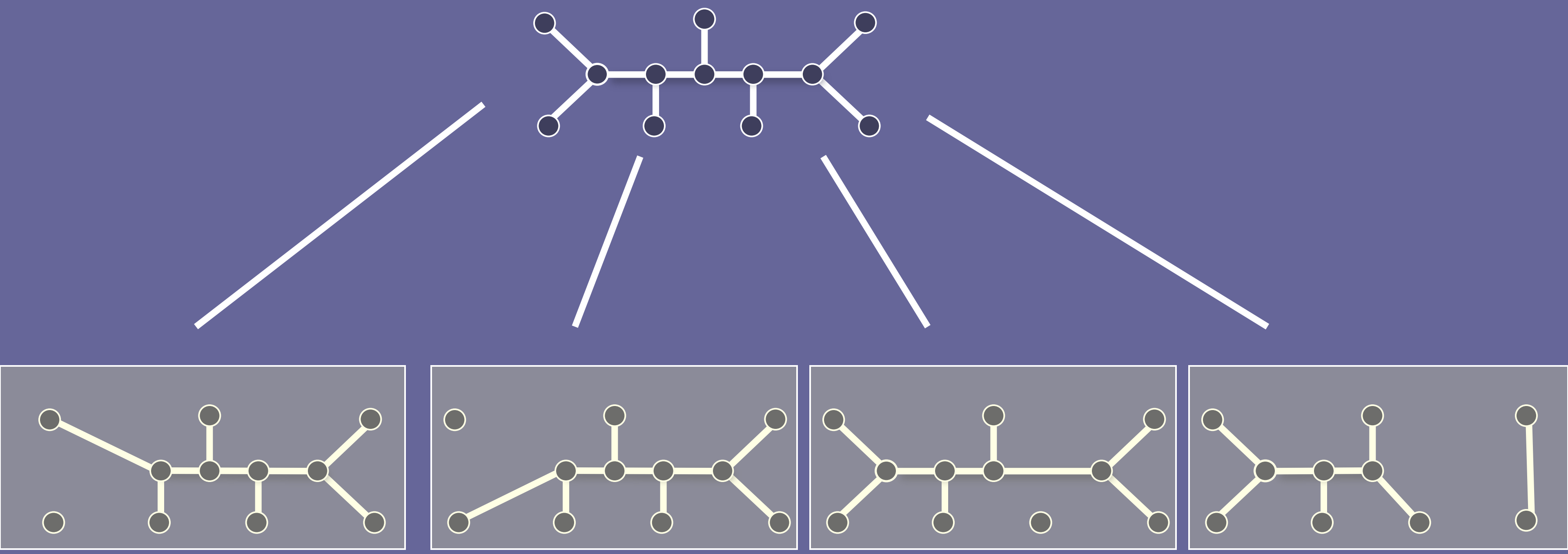
Step 0: F <- dequeue U’ (initially containing only $T_1$)

Step 1: If there is a min incompatible quartet Q in F wrt $T_2$,
and $|F| \leq k$, then
for each prong of fork R induced by Q in F
construct a new forest F’ obtained by the depronging,
enqueue each of the four forests in U’.
If F is compatible with $T_2$ then add F to U.

Step 2: If U’ empty stop, else go to Step 0.

Output: A collection U of forests compatible with $T_2$, with each such forest F having $|F| \leq k+1$.

# FPT algorithm: Phase I



# $\beta$-mapping

$\beta$ : V(F) -> V($T_2$)

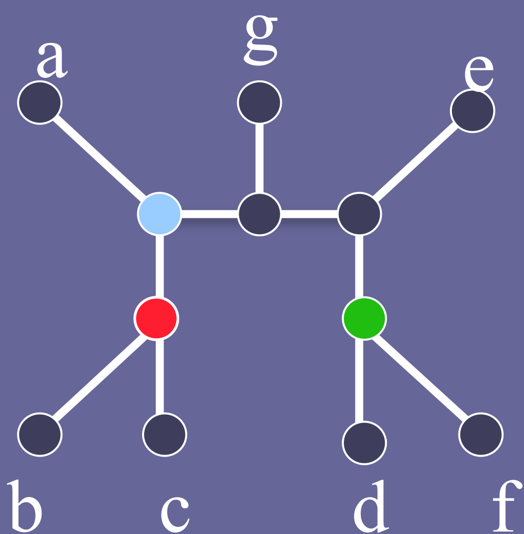
*leaves* in F map to leaves in $T_2$ with matching labels

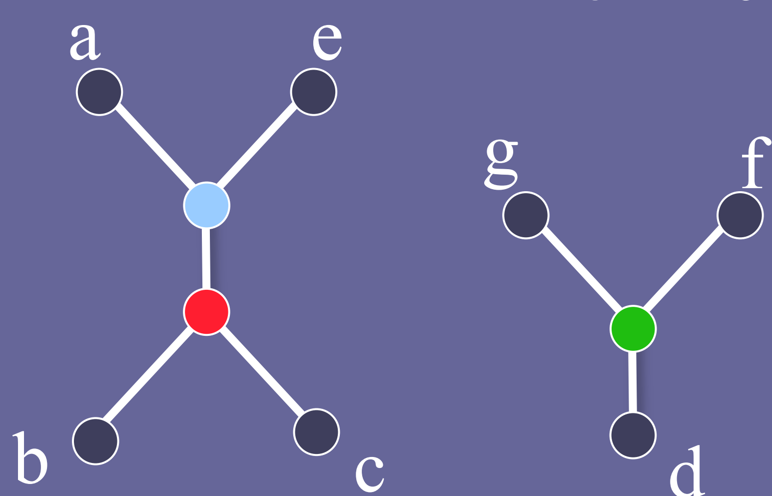
for an *internal vertex* u :
let $T_1$, $T_2$, $T_3$ be the three subtrees obtained from F \ u and
let $l_1$, $l_2$, $l_3$ be leaves in $T_1$, $T_2$, $T_3$ resp.
u maps to the unique vertex in $T_2$ that is on all three paths
$P_{F2}(l_1, l_2)$, $P_{F2}(l_2, l_3)$, $P_{F2}(l_1, l_3)$,

If F and $T_2$ are compatible, then $\beta(.)$ is well-defined.

# β-mapping

$T_2$

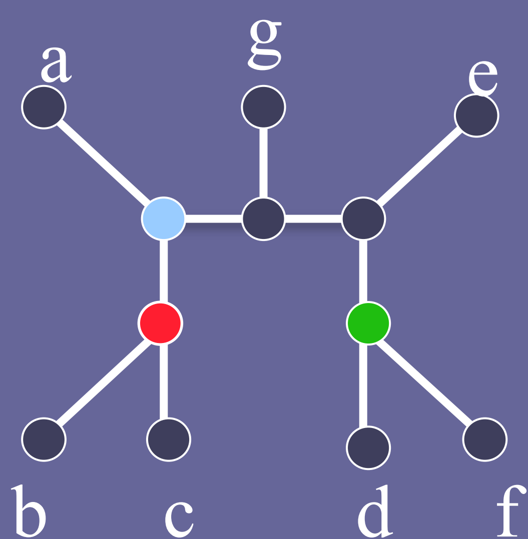


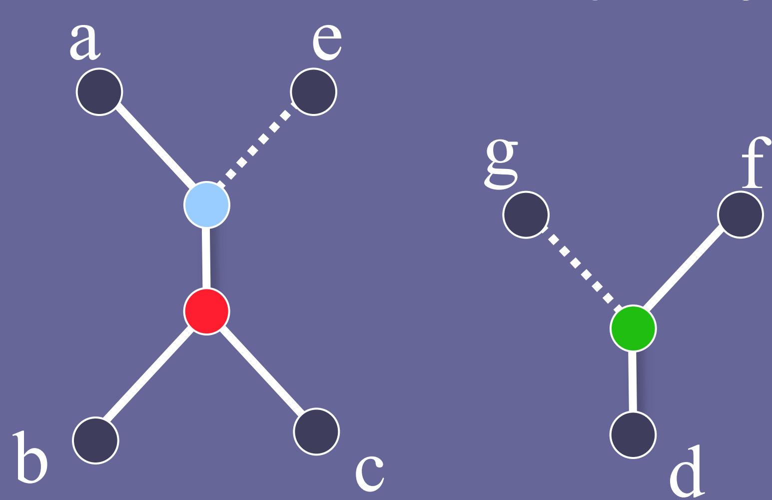




F

# Obstructions

T$_2$




F

dashed edges are *obstructions* for F and T$_2$

# Obstruction property

(a,b) and (c,d) are obstructions for F and $T_2$ if
$P_{T2}(\ \beta(a), \beta(b))$ and $P_{T2}(\ \beta(c), \beta(d))$ share a vertex.

If F and $T_2$ are compatible, then obstructions (a,b) and (c,d) are in separate components of F, say $t_i$ and $t_j$ resp.

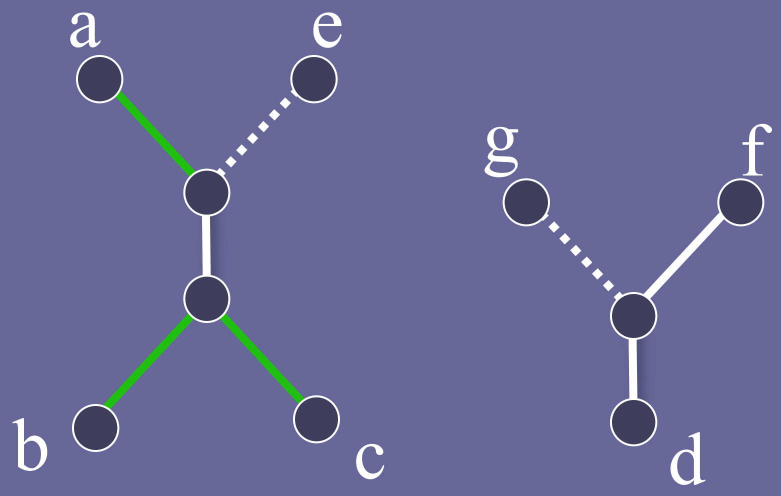
Let $T_a$, $T_b$, be the two trees produced from $t_i$ by removing (a,b), $T_c$, $T_d$ be the two trees produced from $t_j$ by removing (c,d).

Consider any set of leaves $\{l_a, l_b, l_c, l_d\}$ with $l_x$ in $T_x$.
In any MAF $F = \{t_1, t_2, \ldots t_k\}$ for F and $T_2$ ,
no sets of the form $\{l_a, l_b\} \subseteq t_i$ *and* $\{l_c, l_d\} \subseteq t_j$, for any $1 \leq i,j \leq k$.

# Obstruction property

Let e and e' be obstructions for F and $T_2$.
Let *F'* be an agreement forest for F and $T_2$.

Any set of edges such that $\tau(F \setminus E)$ produces *F'* from F will either contain one of {e, e'}, or an edge that can be exchanged for one of {e, e'} to produce E', such that $\tau(T_1 \setminus E')$ produces *F'* from $T_1$.




F

# FPT algorithm: Phase II

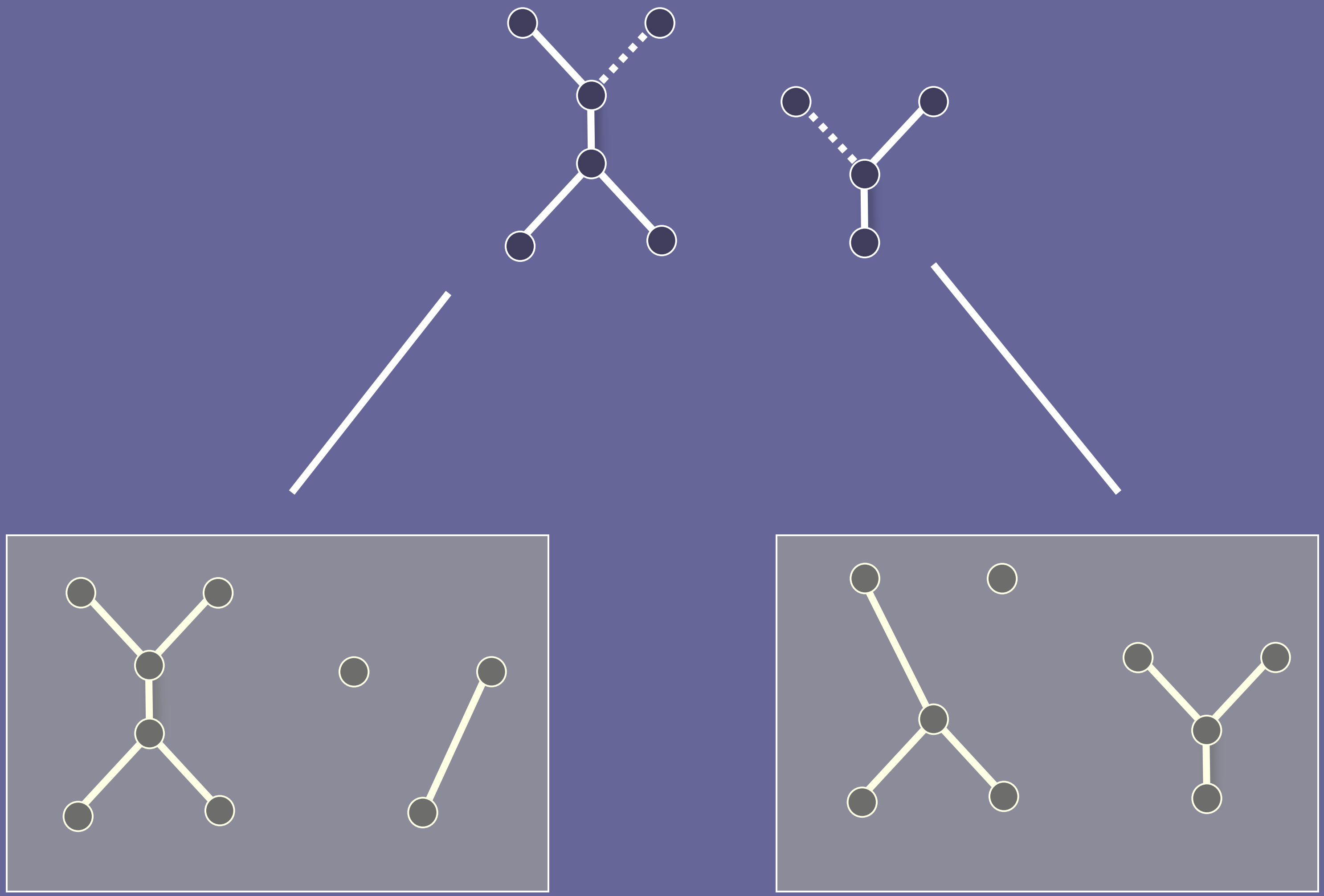
Step 0: F <- dequeue U’ (initially the output queue from Phase I)

Step 1: If there are obstructions {e, e’} in F wrt $T_2$, and $|F| \leq k$,
then, for each of e and e’
construct new forests $\tau(F \setminus e)$ and $\tau(F \setminus e')$,
enqueue these two forests in U’.
If no obstructions exist in F wrt $T_2$ then add F to $U_{final}$

Step 2: If U’ empty stop, else go to Step 0.

Output: A collection $U_{final}$ of forests that are agreement forests for $T_1$ and $T_2$, with each such forest F having $|F| \leq k+1$.

# FPT algorithm: Phase II



# Algorithms?

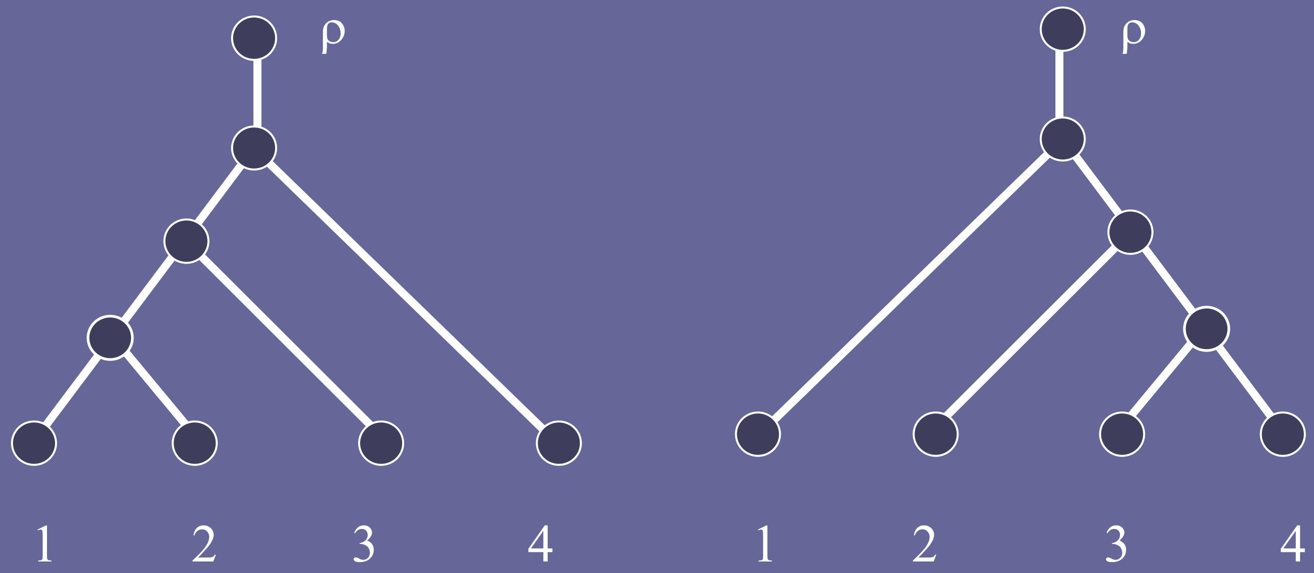
FPT algorithm for parameterized k-TBR problem
running time $O(4^k \cdot k^5) + p(|X|)$ (when combined with kernelization.)

We get for free:

4-approximation algorithm for TBR problem, running time $O(|X|^6)$.
Throw in all prongs at each step in Phase I, both obstructions at each step in Phase II.

Parameterized approximation algorithm for k-SPR problem
Output: Either a set of SPR operations $T_1$ -> $T_2$ of size at most 2k
or 'NO' ( only if SPR distance for $(T_1,T_2)$ is > k )

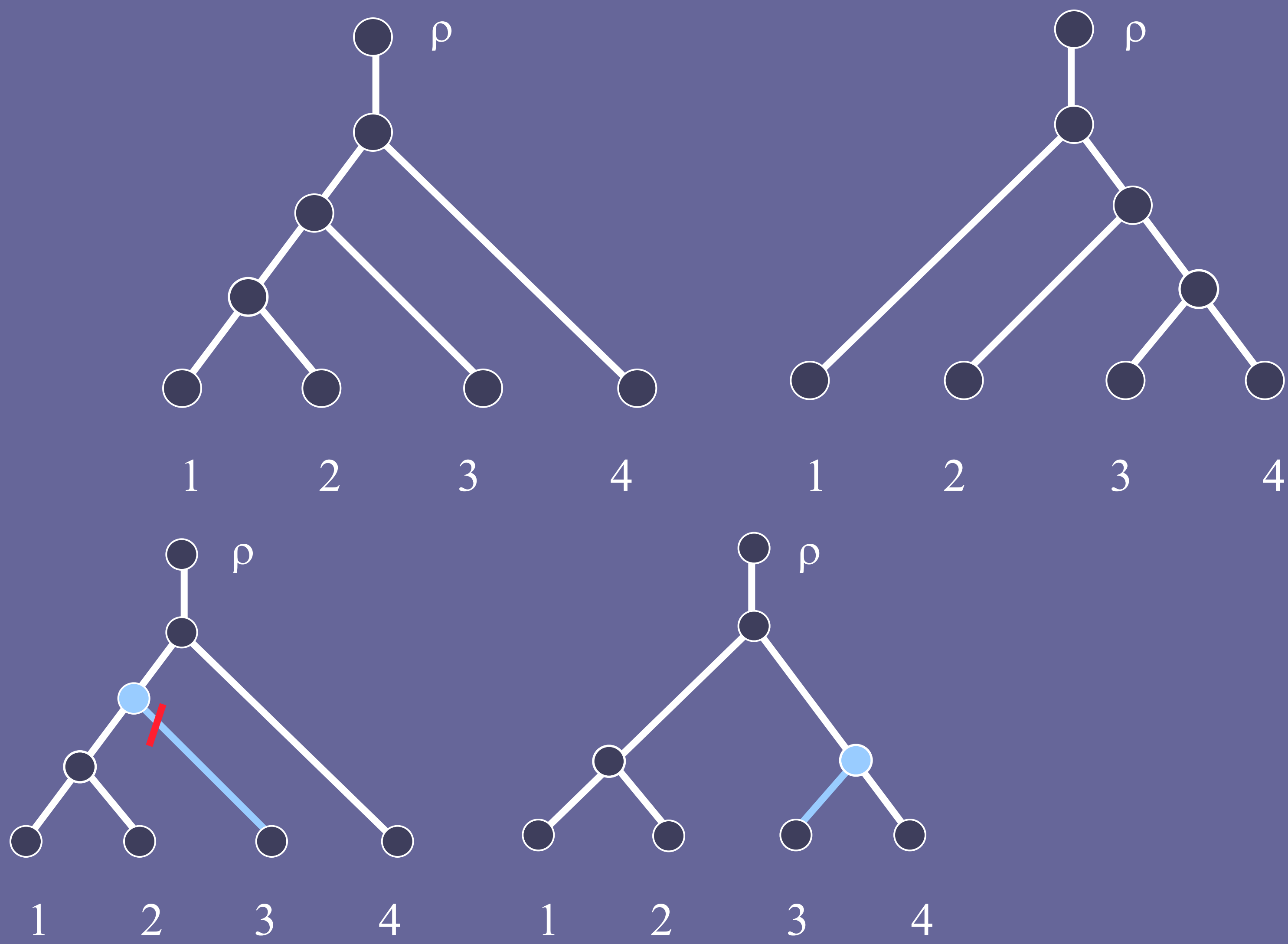
# What about rooted trees?



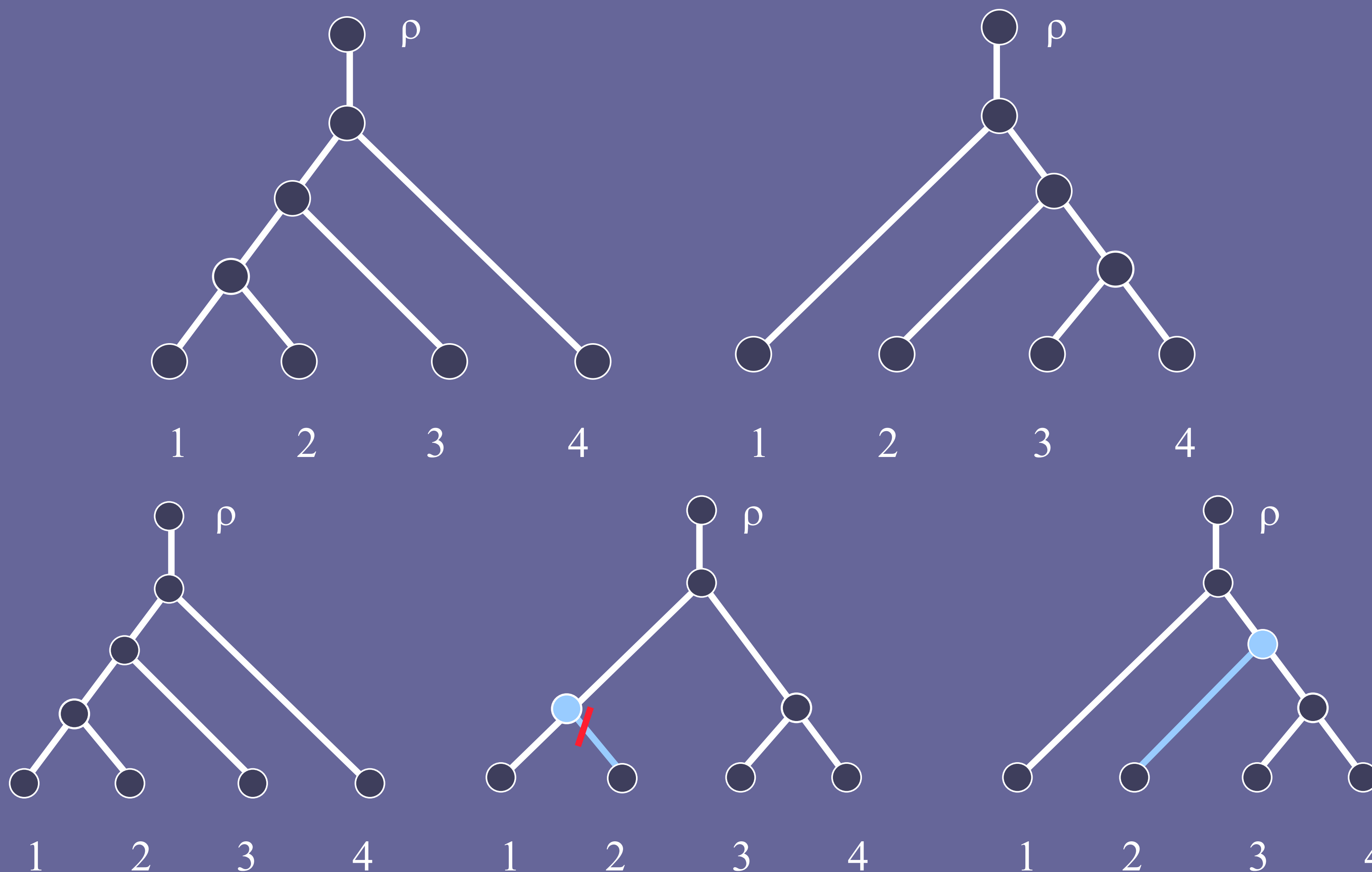

using *planted* rooted trees, we can characterize rSPR distance in terms of agreement forests.

(Hein et al. 1996, Bordewich and Semple 2004)

# rSPR distance for rooted X-trees



# rSPR distance for rooted X-trees



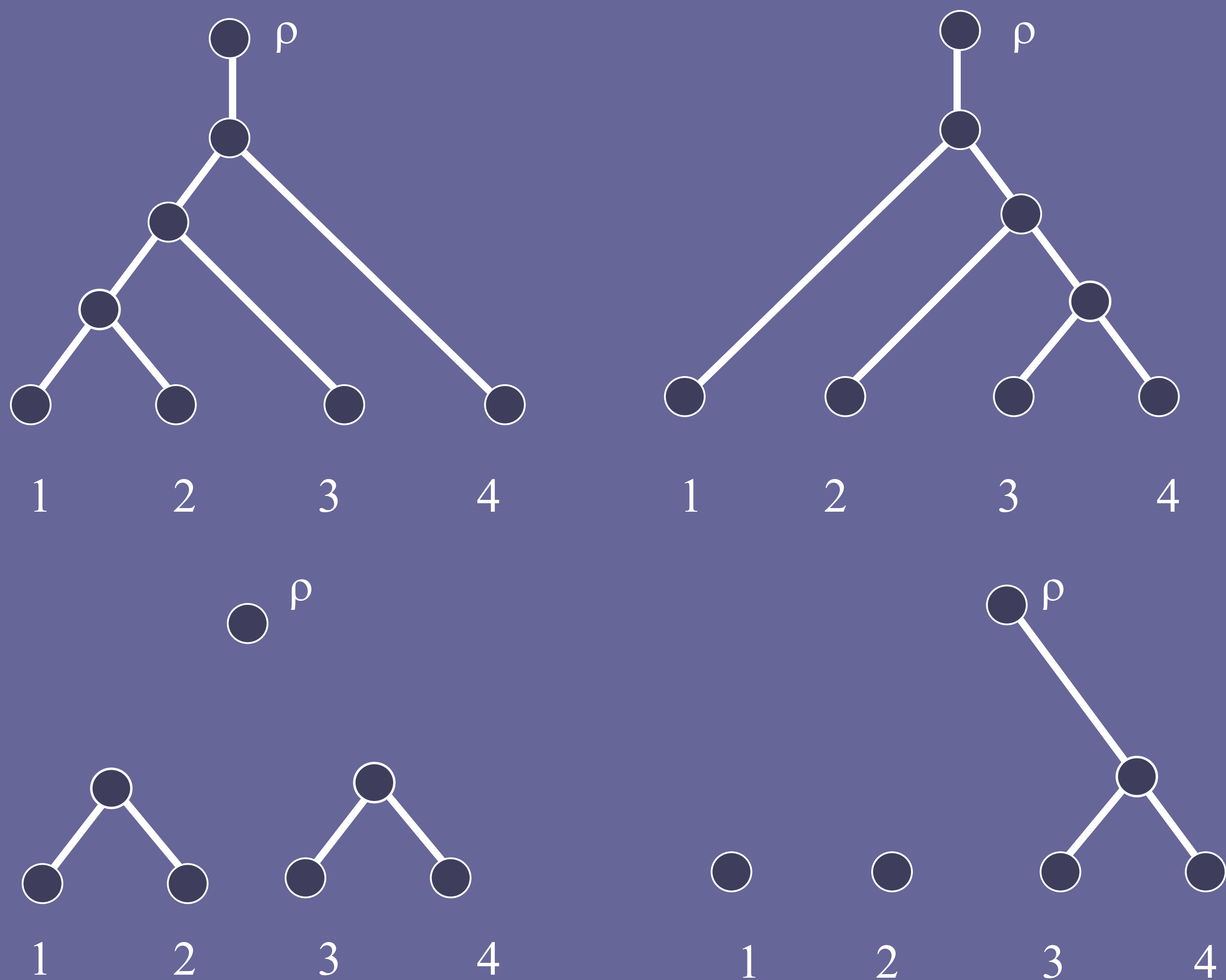
# Agreement forest for rooted X-trees $T_1$ and $T_2$

$F = \{t_\rho, t_1, t_2, \ldots t_k\}$ a collection of phylogenetic trees such that

1. $L(t_\rho), L(t_1), \ldots, L(t_k)$ partitions $X \cup \{\rho\}$
2. $\forall j \in \{\rho, 1,\ldots, k\} \quad t_j = \sigma(T_1[L(t_j)]) = \sigma(T_2[L(t_j)])$
3. for $i = 1, 2$ the trees $\{T_i[L(t_j)] : j \in \{\rho, 1,\ldots, k\}\}$ are vertex disjoint subtrees of $T_i$

$\sigma$ (.) and $\tau$ (.) defined as before except component roots are preserved

# Agreement forests for $T_1$ and $T_2$



## Maximum agreement forest (rMAF) for rooted trees $T_1$ and $T_2$

an agreement forest $F$ for $T_1$ and $T_2$ with $|F| = k$ minimized

$k\text{-}1 = m_r(T_1,T_2)$ is equal to the rSPR distance for $T_1$ and $T_2$
computing $m_r(T_1,T_2)$ is NP-hard

(Hein et al. 1996, Bordewich and Semple 2004)

k-rMAF Problem:
Input: Pair of planted rooted phylogenetic X-trees $T_1$ and $T_2$
Parameter: k
Output: Agreement forest $F = \{t_\rho, t_1, t_2, \ldots t_{k'}\}$ for $T_1$ and $T_2$
with $k' \leq k$, or 'NO' if $m_r(T_1,T_2) > k$

# FPT algorithm for rSPR:

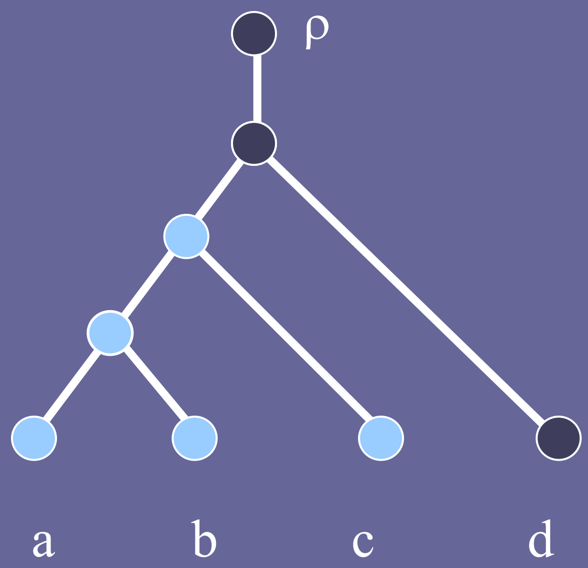
1st attempt, again use *kernelization.*
$(T_1, T_2, k)$ reduced to $(S_1, S_2, k)$ with $|L(S_{1)}| = |L(S_2)| = |X'| < 28k$.

at most $4|X'|^2$ possible single rSPR operations
examine all possible paths of length k from $S_1$
in time $O((56k)^{2k}) + p(|X|)$

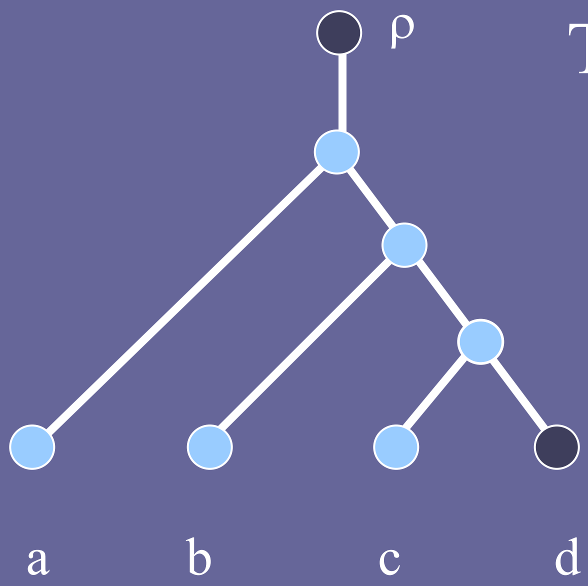
(Bordewich and Semple 2004)

# Incompatible triple

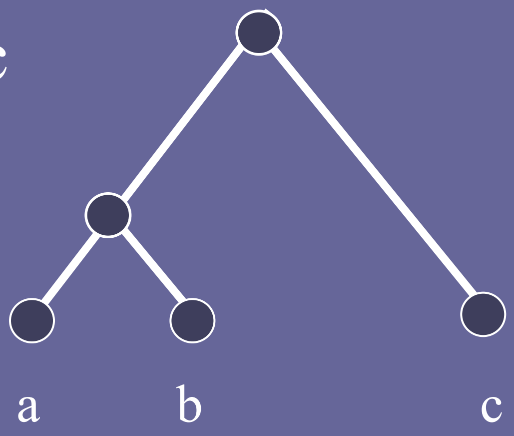
F



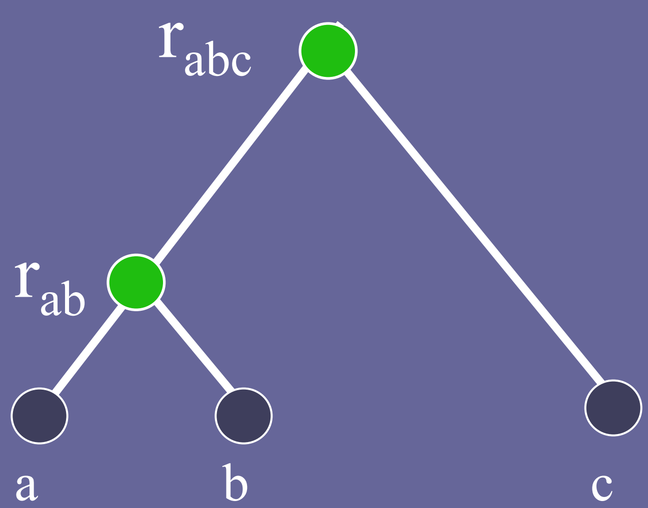

$T_2$




R = ab | c



# Minimal incompatible triple




define a partial ordering on triples:

$xy \mid z < ab \mid c$ if $r_{xyz}$ is a descendant of $r_{abc}$

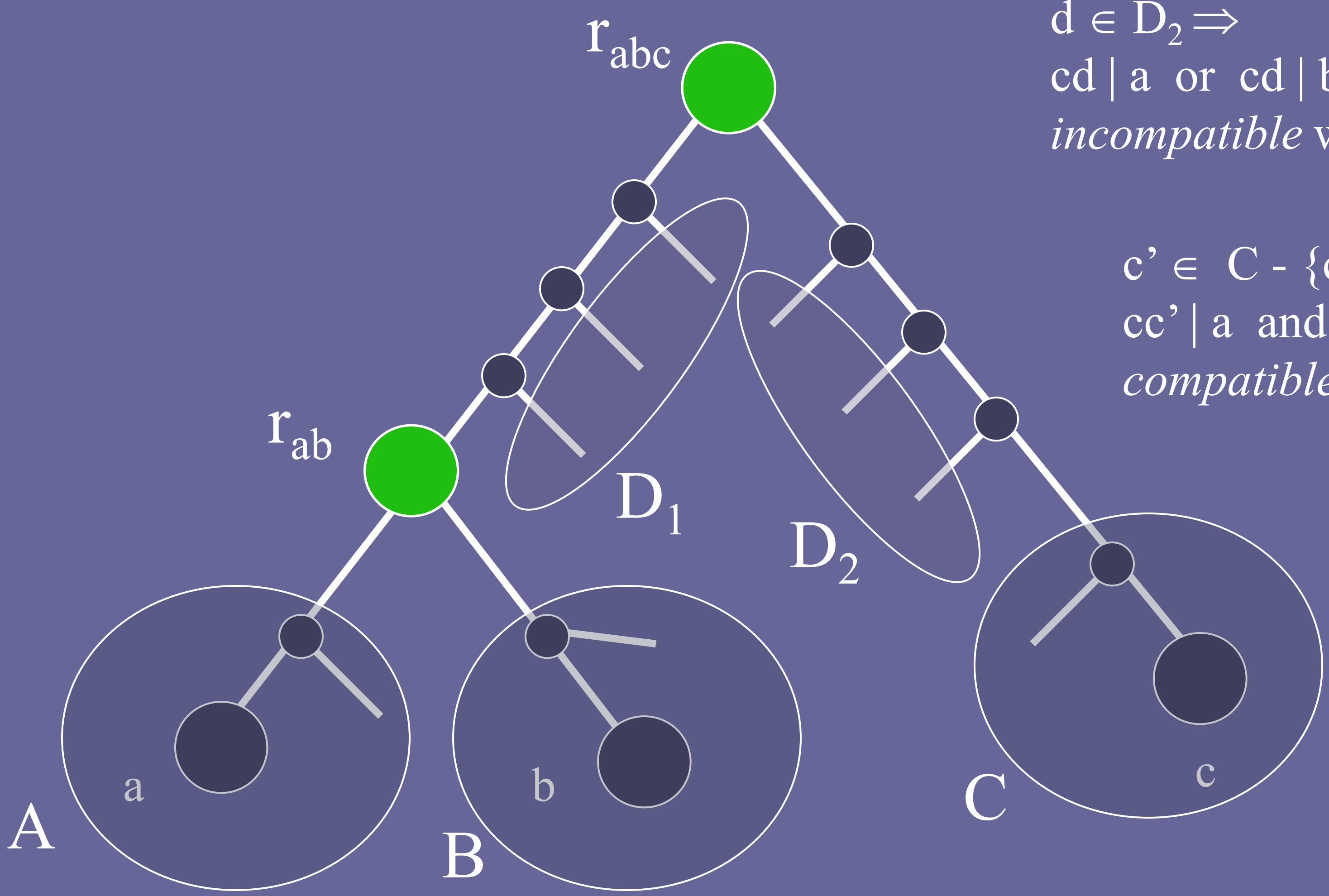
or, $r_{xyz} = r_{abc}$ and $r_{xy}$ is a descendant of $r_{ab}$

$R = ab \mid c$ in F is a *minimal incompatible triple* wrt F and $T_2$

if 1. R is incompatible with $T_2$

2. no $R' < R$ in F incompatible with $T_2$

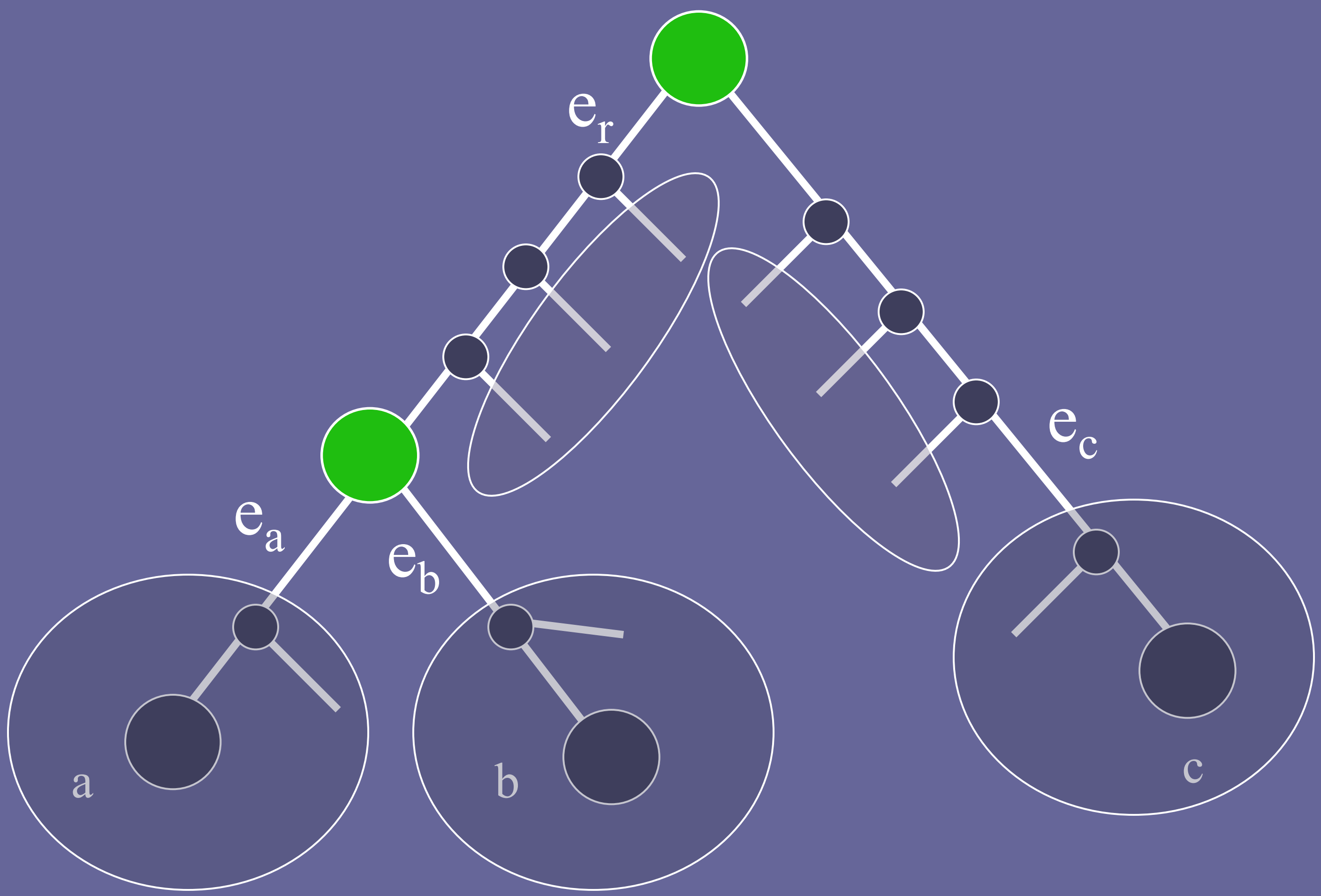
# Layout of a minimal incompatible triple in F




$d \in D_2 \Rightarrow$
$cd \mid a$ or $cd \mid b$
*incompatible* with T2

$c' \in C - \{c\} \Rightarrow$
$cc' \mid a$ and $cc' \mid b$
*compatible* with T2

# Layout of a minimal incompatible triple in F

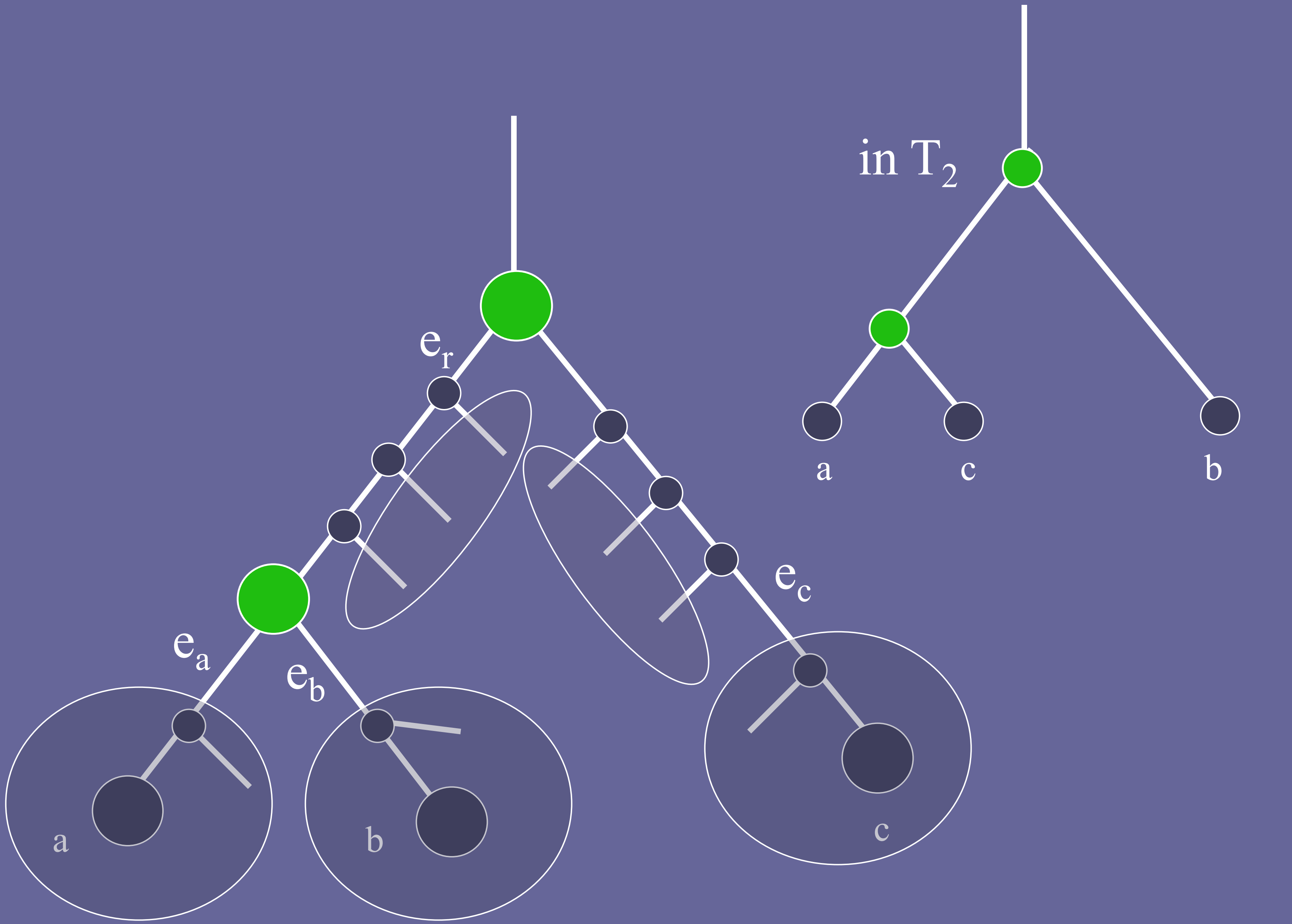


# Minimal incompatible triple property

$R = ab \mid c$ , a minimal incompatible triple in F wrt $T_2$,
with edges $\{e_a, e_b, e_c, e_r\}$ as shown in the layout.

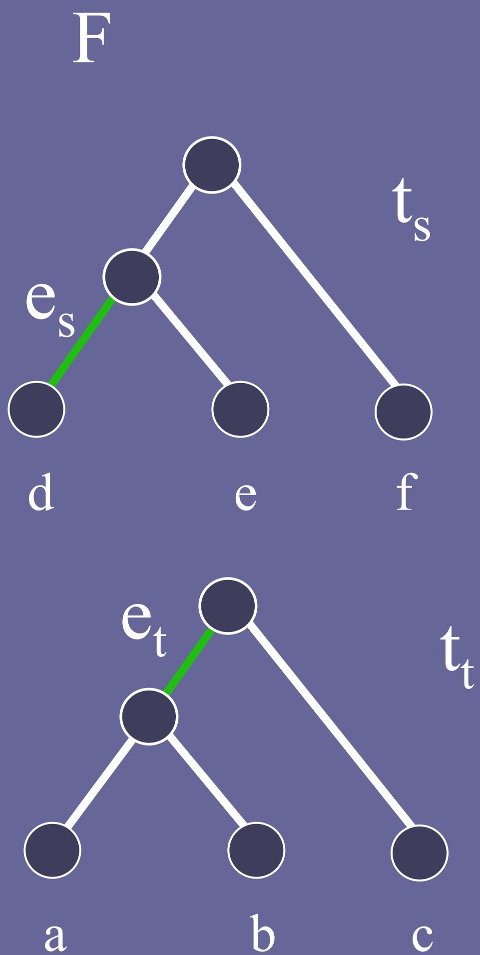
let *F'* be an agreement forest for F and $T_2$.

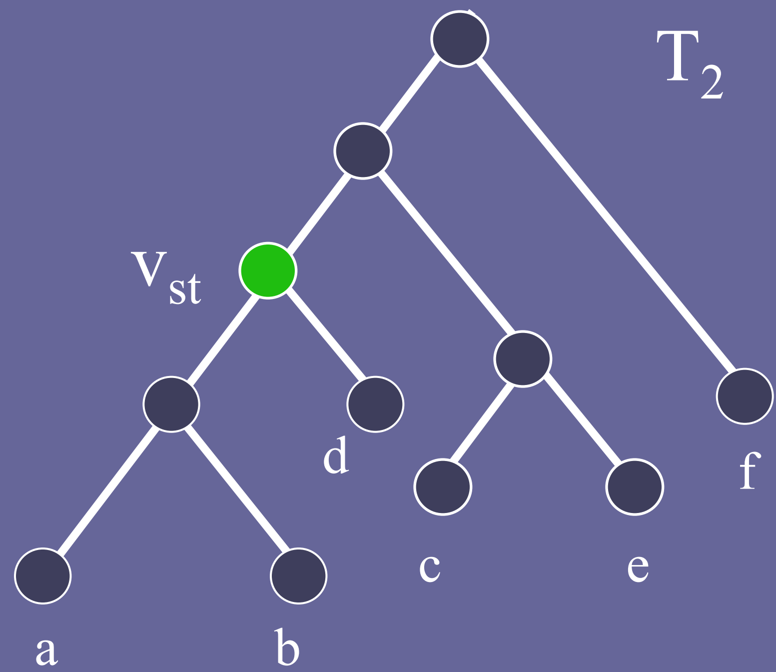
any set of edges E such that $\tau(F \setminus E)$ produces *F'* from F will either contain one of $\{e_a, e_b, e_c, e_r\}$, or an edge that can be exchanged for one of $\{e_a, e_b, e_c, e_r\}$ to produce E', such that $\tau(T_1 \setminus E')$ produces *F'* from $T_1$.



in $T_2$
$e_r$
$e_a$
$e_b$
$e_c$
a
b
c
a
c
b

# Overlapping components



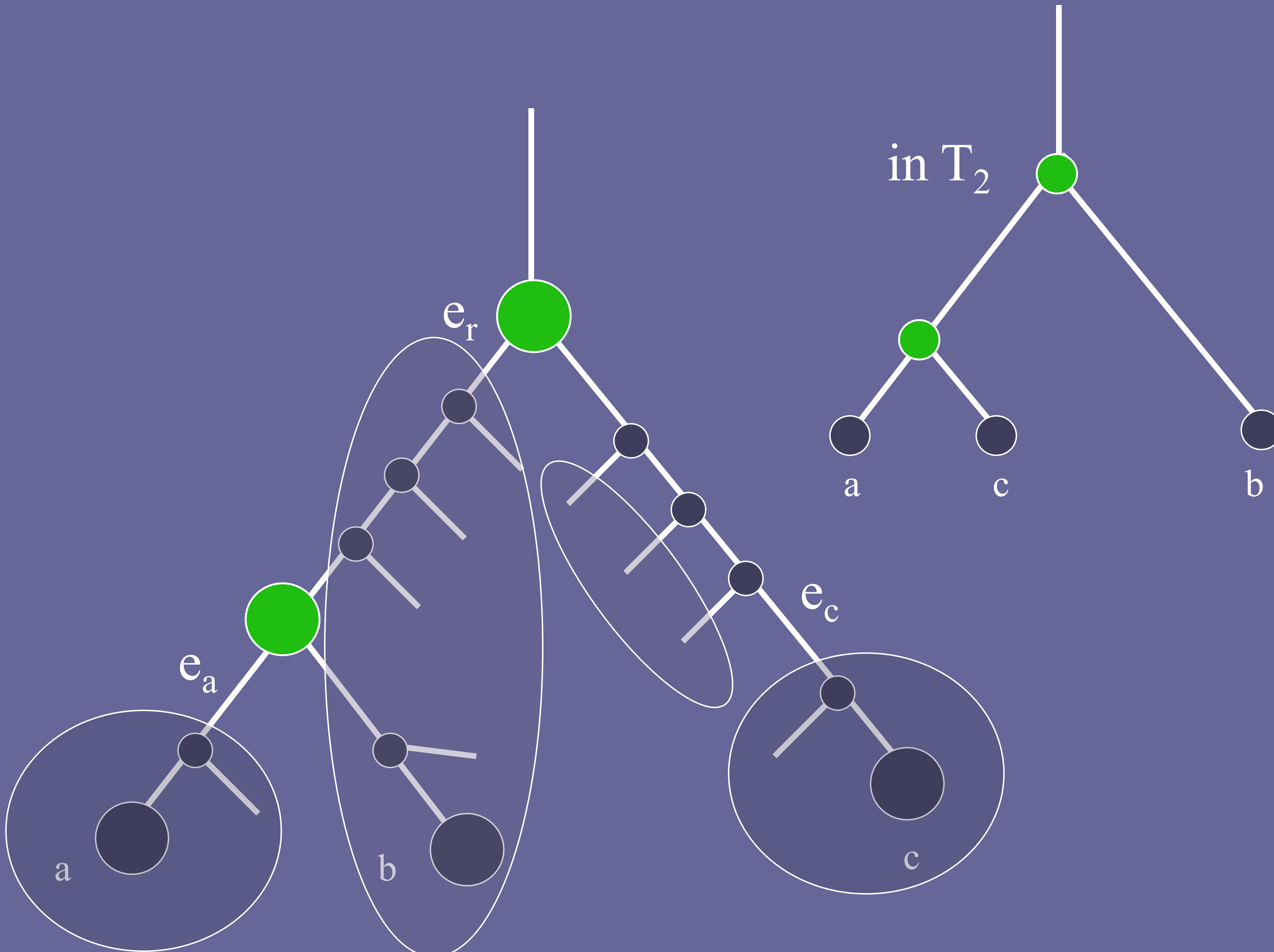

given $t_s$, $t_t$, overlapping in $T_2$,
choose $v_{st}$ minimal in $T_2$
$e_s$ and $e_t$ are obstructions in F

# A 3-approximation algorithm for rSPR

R = ab | c , a minimal incompatible triple in F wrt $T_2$, with edges $\{e_a, e_b, e_c, e_r\}$ as shown in the layout
let *F'* be an agreement forest for F and $T_2$.

any set of edges E such that $\tau(F \setminus E)$ produces *F'* from F will contain an edge that can be exchanged for the set $\{e_a, e_c, e_r\}$ to produce E', such that $\tau(T_1 \setminus E')$ is a subforest of *F'*

Main point: we can ignore $e_b$



in $T_2$
$e_r$
$e_a$
$e_c$
a
b
c
a
c
b

# rSPR Results

FPT algorithm for parameterized k-rSPR problem
running time $O(4^k . k^4) + p(|X|)$

3-approximation algorithm for rSPR problem, running time $O(|X|^5)$

previous best: 5-approximation, linear time
(Bonet, St John, Mahindru 2006, based on Hein et al. 1996, Rodrigues et al. 2001)