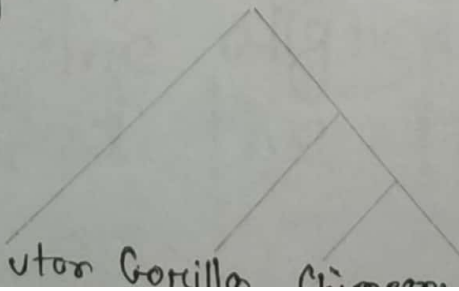
Phylogetic tree

Phylogenetic tree / Phylogeny:

A phylogenetic tree also known as a phylogeny that depicts the line of evolutionary descent of different species, organisms or genes from a common ancestor.

- The origin and evaluation of species has always been a mystery & fascinating research area.
- represents the evolutionary history of a group of organisms.



NB:

* every internal node → hypothetical ancestor
* every leaf node → species / taxa
*

When to say two individuals are different species?

Reproduction between two different species is not possible.

Phylogeny reconstruction:

Phylogenetic tree can be constructed from

1. Morphological data
2. Biomolecular data/sequences

* tree can be constructed from various types of data
  - → distance based
  - → character ,,
  - → DNA/protein sequences
  - → gene order

Morphological data:

→ Form and structure of organisms and their specific structural features.

→ number of legs, color of the eye etc.

Biomolecular data/sequences

DNA, RNA, amino acids etc.

Steps to construct phylogenetic tree (from sequence)

* acquiring a set of homologous DNA or protein sequences
* align those sequence (MSA)
* construction of phylogenetic tree from the aligned sequences
* present the tree.

Applications of phylogenetic tree:

- → gene function identification
- → infer the amino acid sequence of extinct proteins
- → track the evolution of disease
- → investigation of criminal case

Gene tree: A phylogenetic tree that depicts how a single gene has evolved in a group of related species

Species tree

Pattern of branching of species lineages via speciation. It shows the relationship between

different species.

Discordance:

Gene tree don't necessarily show the same branching pattern as their containing species tree.

Reason for gene tree discordance

→ Deep coalescence or Incomplete Lineage Sorting (ILS)

→ Gene duplication/Extinction

→ Horizontal Gene transfer (HGT)

→ Ex Estimation errors may also introduce discordance

Speciation (স্পেসিয়েশন)

Coalescent point

Point in the past in which all gene copies arise from a single gene copy.

Deep coalescence / Incomplete lineage sorting

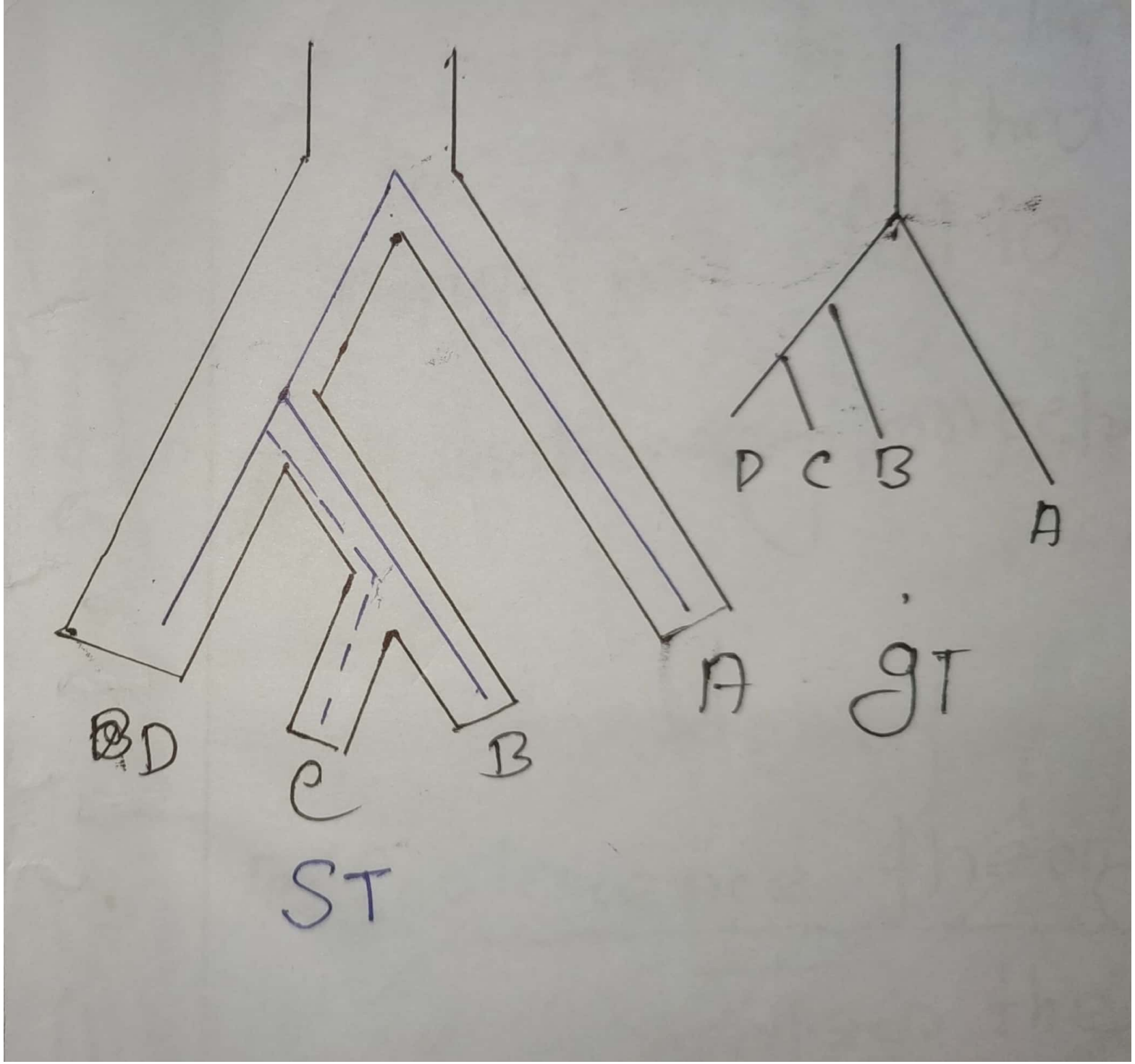
Gene copies fail to coalesce in the speci-ation point.

Gene copies at a single locus extends deeper than the speci-ation events.

things that influence the coalescent points

Population size:

* Smaller population size and longer branches increases the


D
C
B
A
D
C
B
A
ST
GT

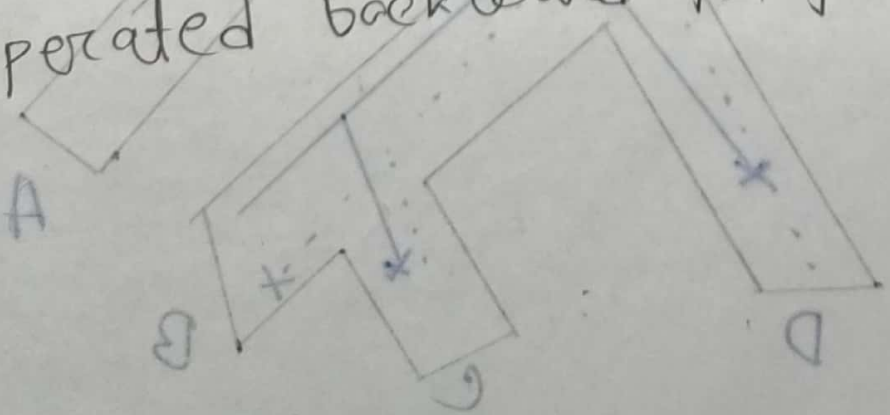
possibility of coalescence

* Larger population size and shorter branches increase the chances that gene copies will fail to coalesce

2. Evolutionary mechanisms (drift, type of selection)

Coalescence theory

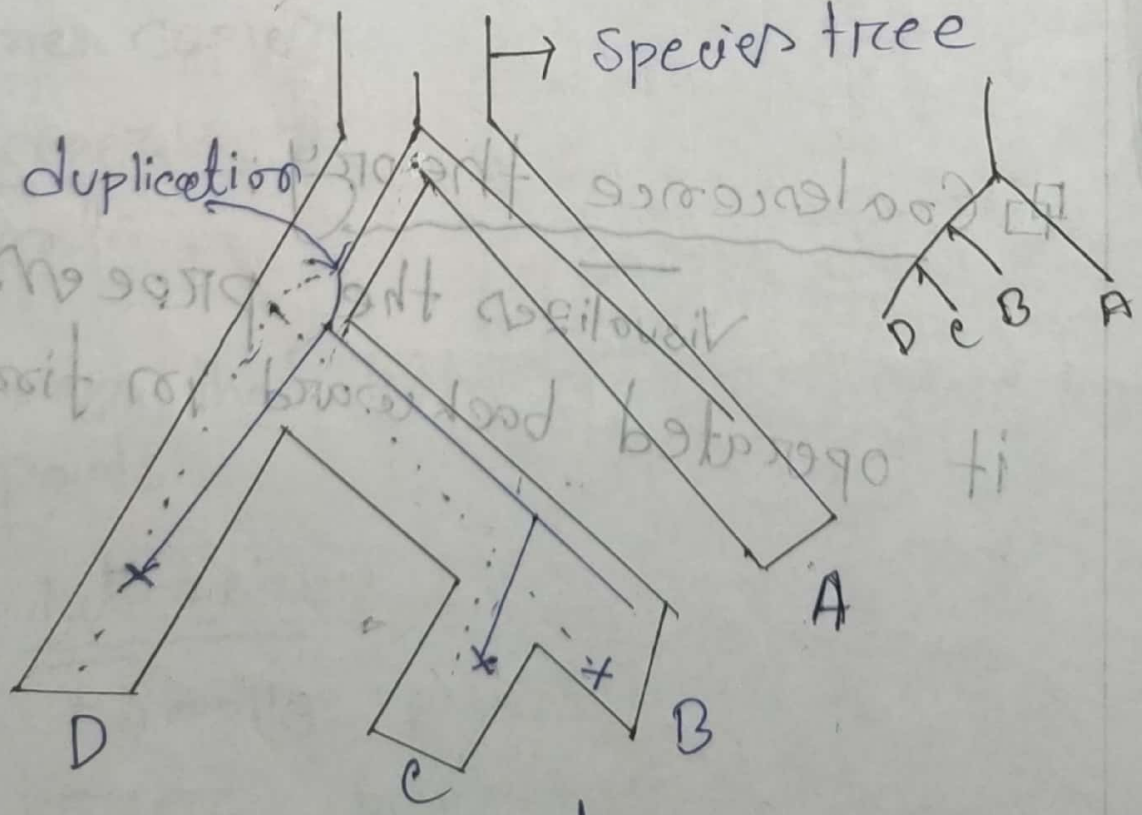
Visualizes the process as if it operated backward in time



Gene duplication / loss:

A gene might get duplicated and both copies descend and evolve independently.

Discordance can occur if some sampled copies come from one locus and others come from another locus.



1 duplication and 3 losses

Confine: limit/space

Horizontal gene transfer

Renegade genes somehow break the confines of the species lineages and moved horizontally across the phylogeny

NB: network instead of tree

Reconciliation:

An approach to connect the history of two or more coevolving biological entities.

A phylogenetic tree representing the evolution of an entity can be drawn within another phylogenetic tree representing encompassing entity to reveal their interdependence and the evolutionary events.

⊞ Most recent common ancestor (MRCA) /

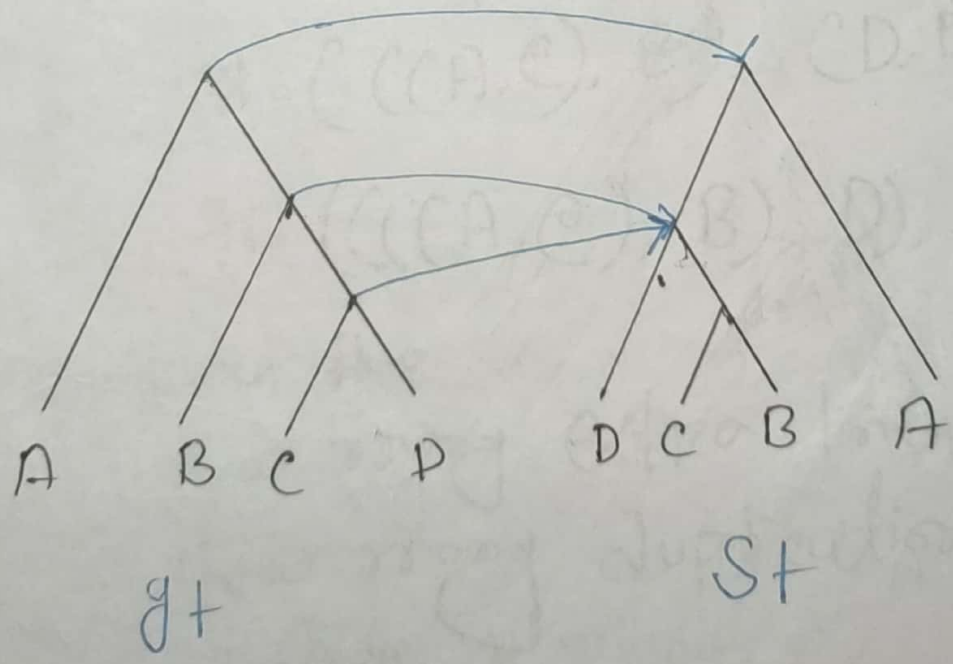
** Last common ancestor (LCA)

MRCA/LCA mapping (gt → St)

* An algorithm to find optimal reconciliation:

→ find number of duplicate (minimum)

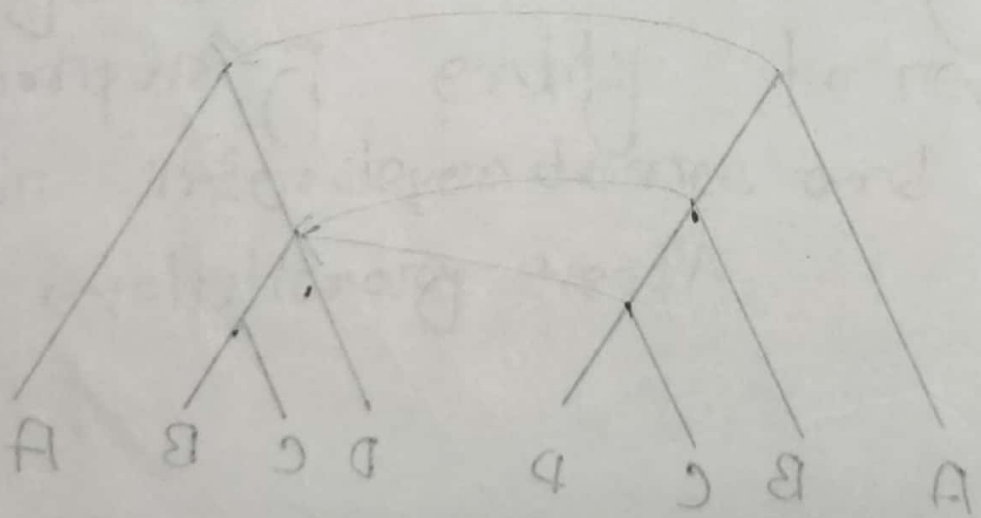
→ find the location of duplicate



theorem / duplication node:

An internal node v of gt is a duplication node if and only if $M(v) = M(w)$ for some child w of v.

mapping of v



Consider the following gt and st pairs of Specification.

P-1 gt = ((CB, D), C), A)
st = ((CA, B), C), D)

Newick notation

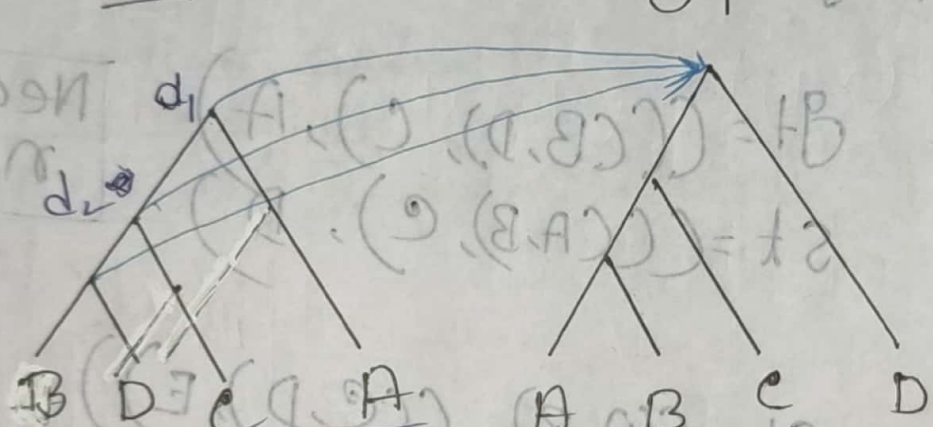
P-2 st = (A, B), C(C, D), E))
gt = ((A, C), (D, E), B))

P-3 st = ((A, C), B), (D, E))
gt = (((A, C), B), D), E))

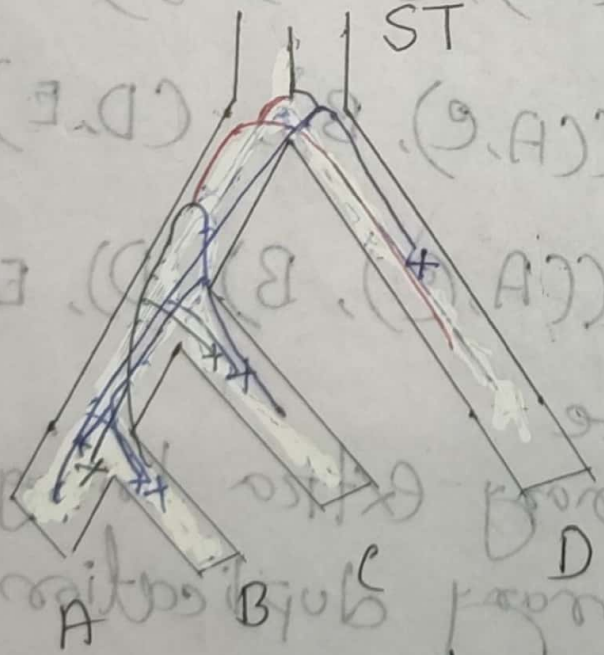
Find out, the
How many extra lineages
How many duplications and losses.

Soln

P-1 / gt

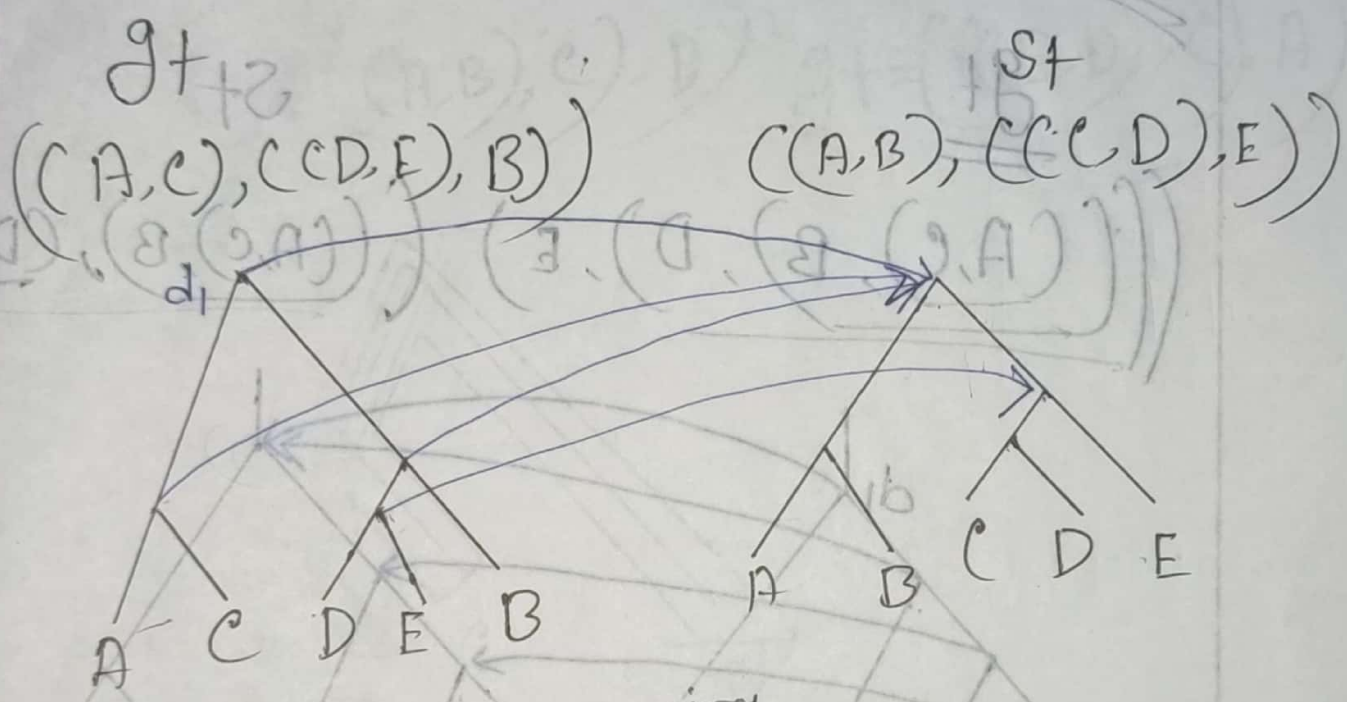


* Here, d_1, d_2 are two duplication points.



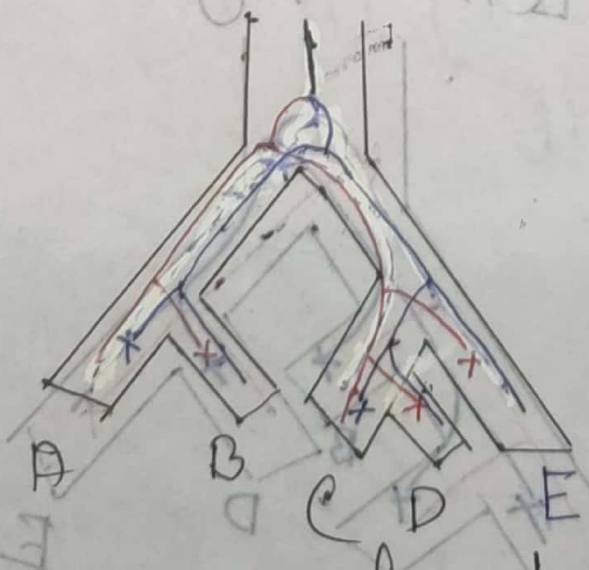
2 duplicate production losses.

P-2



LCA mapping

here d_1 is duplication node



1 duplication ~~to~~ five losses

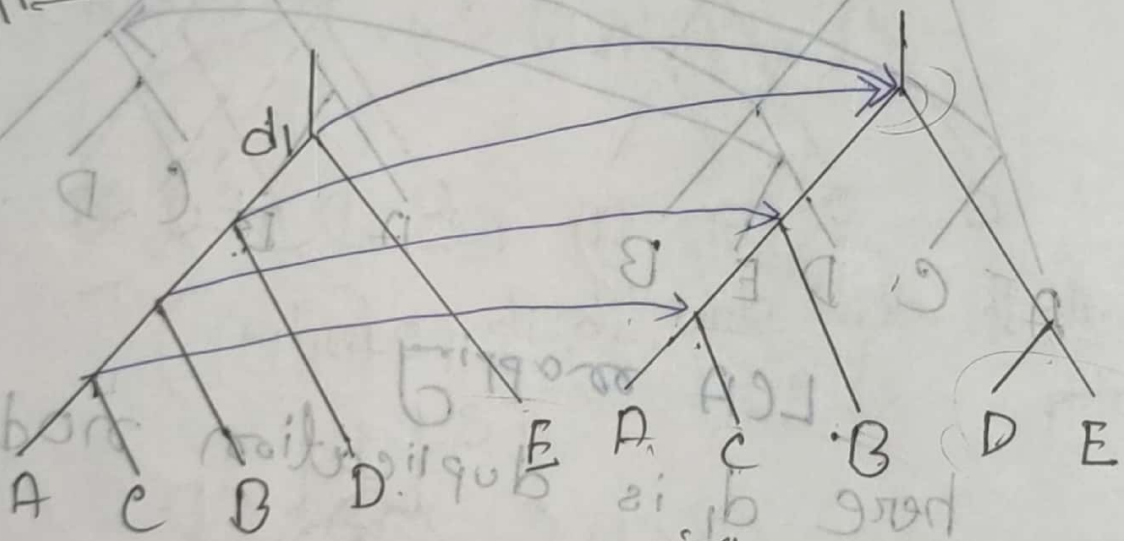
P-3

gt

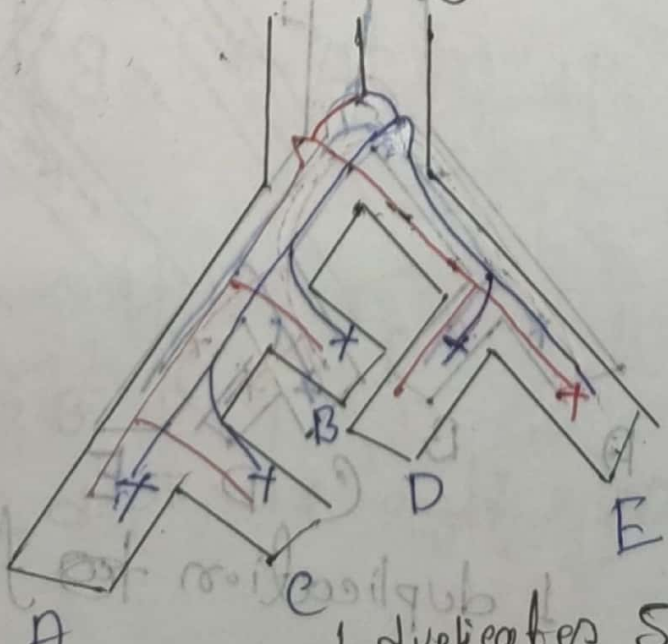
((((A,C),B),D),E)

St

((A,C),B),(D,E))



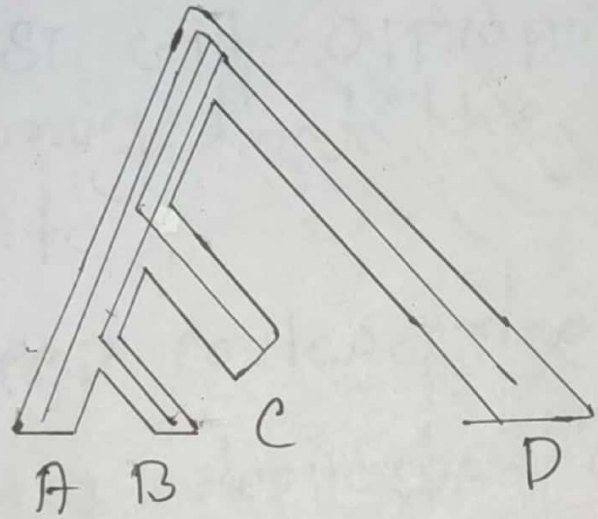
LCA mapping



1 duplications 5 losses

P-1

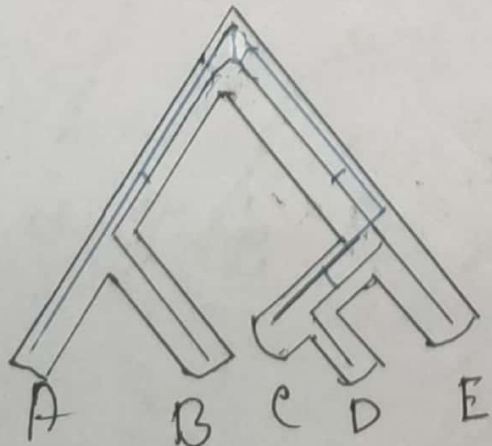
$St = (((A,B),C),D)$ $gt = (((C,B),D),C),A)$



2 extra lineage

P-2 $St = ((A,B),((C,D),E))$

$gt = ((A,C),(CD,E),B))$



3/4 extra lineage

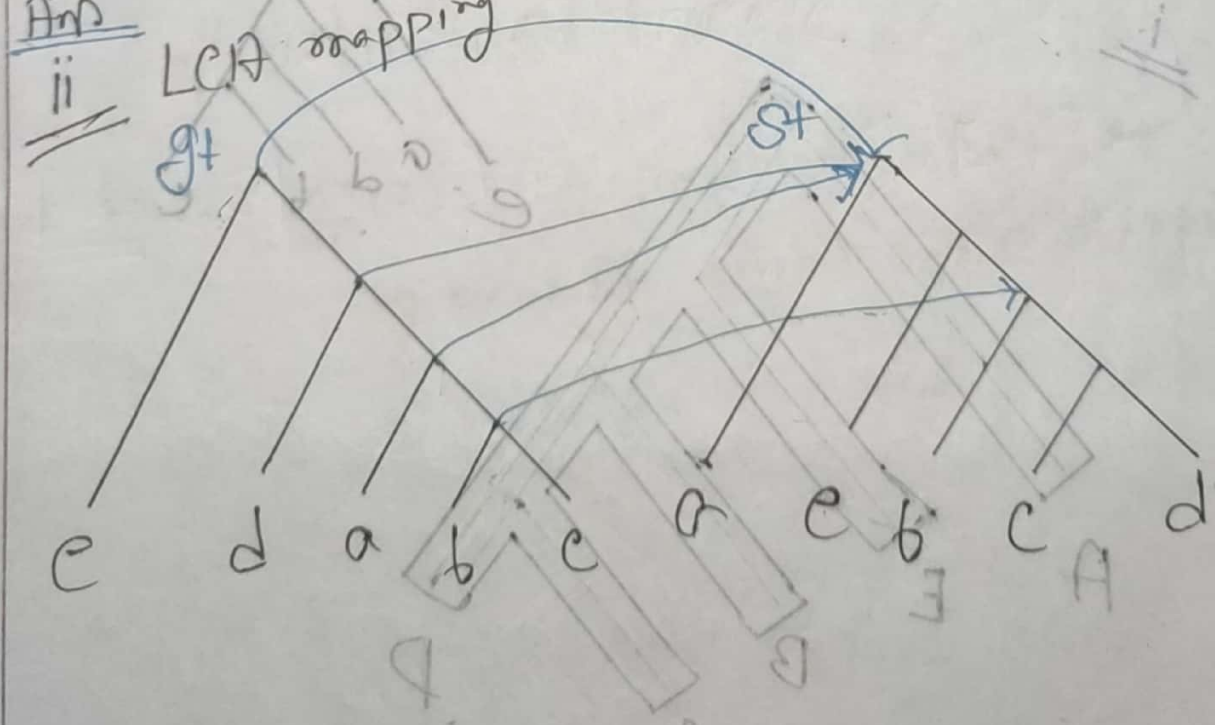
Consider a gene tree $gt = (e,(d,(a,(b,c))))$ and a species tree $ST = (a,(e,(b,(c,d))))$. Explain the discordance between gt and ST with appropriate figure assuming that the discordance is due to

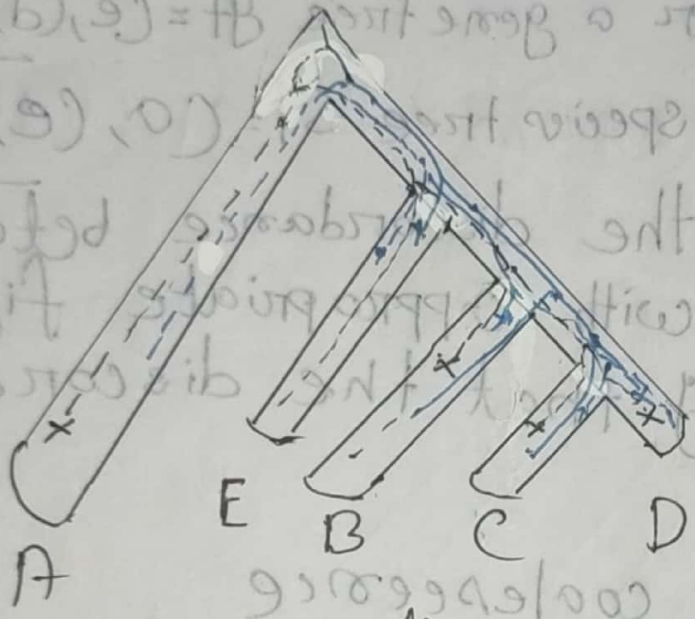
i) Deep coalescence

ii) Gene duplication and losses

Ans

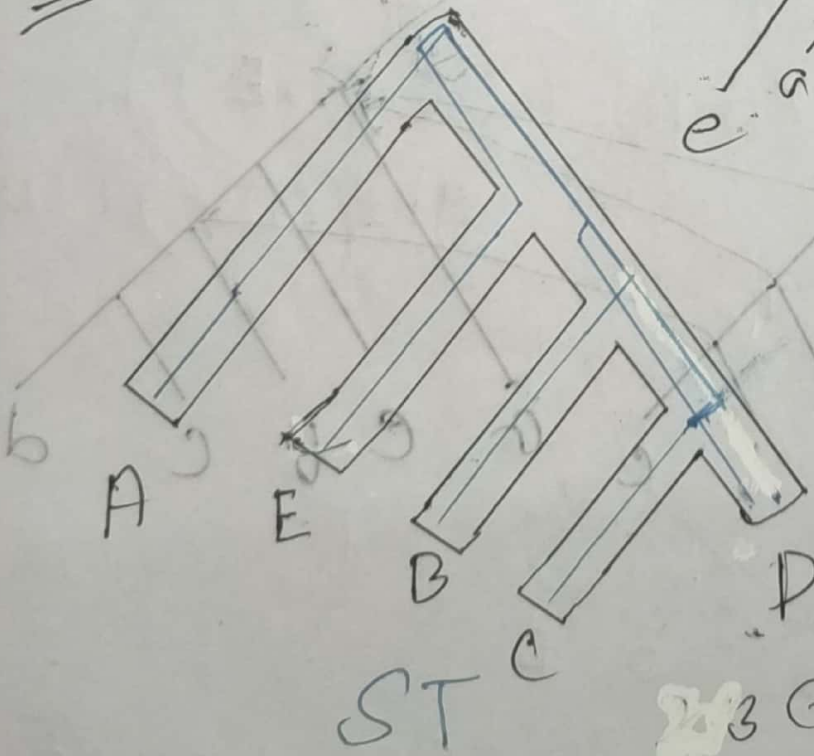
ii LCA mapping





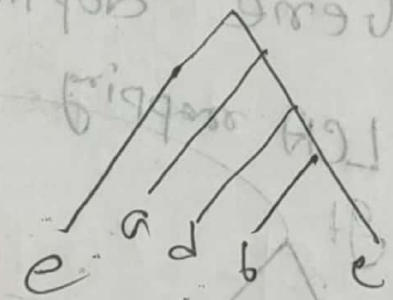
2 duplication

i



ST

ii



3 extra lineage

tree:

Mathematically a tree is a graph $G=(V, E)$, (where V is the vertex set and E is the edge set) that is connected and acyclic

#types of tree:

- rooted tree
- unrooted tree

tree can be represented by

→ Newick representation/notation

→ Clade

Rooted tree:

In a rooted tree T, we can orient the edges in the direction of the root $r = r(T)$, so that all vertices other than r have outdegree one.

* root is denoted by $r(T)$
* set of vertices of T " $V(T)$
* " of edges of T " $E(T)$
* " of leaves of T " $L(T)$

For all nodes $v \neq r$ there is an unique vertex w such that
$v \to w$ is an arc,
w is called the parent of v
v is " the child of w

Siblings: two or more vertex sharing vertices the same parent are called siblings

leaf/tip: A vertex without any children is called a leaf / tip

internal nodes: All the remaining nodes without leaf.

polytomy: A vertex with more than two children

binary - bifurcating or fully resolved: A rooted tree that has no polytomy

multifurcating: A tree that is not binary.

Taxa: In a phylogenetic tree, the leaves represent the taxa of interest.

Ancestor: Internal nodes represent the ancestors of the taxa at the leaves:

Note: There are multiple ways to draw the topology of a rooted phylogenetic tree.

- rotate on top
- rotate on the bottom
- rotate on the left
- rotate on the right

#two way to represent a rooted tree)

Newick notation:

The Newick notation for a rooted binary tree with subtrees A and B is given by (A', B'), where,

A' → newick notation of subtree A

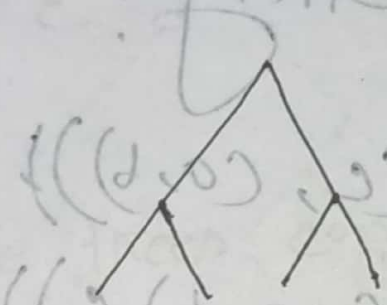
B' → " " of " B

Note:

→ don't care about the left-to-right ordering of subtree; $(A', B') = (B', A')$

→ Newick notation for a leaf is leaf itself

Find Newick notation of following rooted binary tree



((a,b),(c,d))

((a,b),(d,c))

((b,a),(c,d))

((b,a),(d,c))

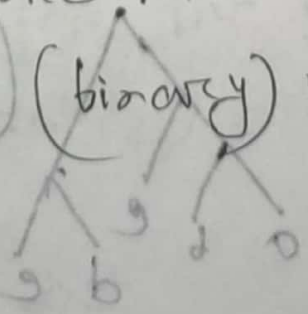
((c,d),(a,b))

((c,d),(b,a))

((d,c),(a,b))

((d,c),(b,a))

NB: there is more than one newick string for a (binary) rooted tree.

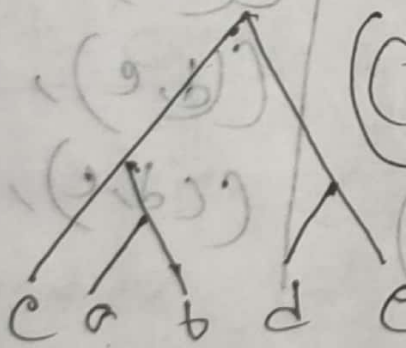


Rooted tree for the following newick string:

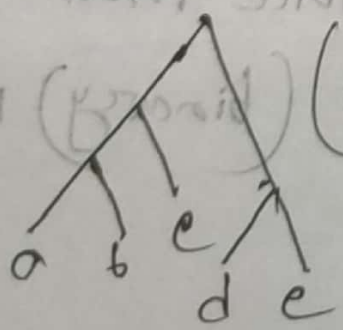
((d,e),(c,(a,b)));

((d,e),((a,b),c));

((e,d),(c,(a,b)));

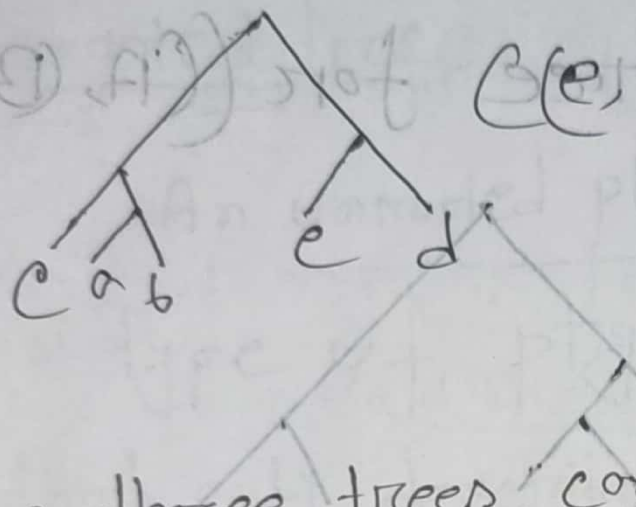


((d,e),(c,(a,b)));



((d,e),(a,b),c));

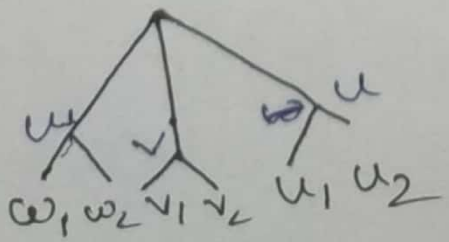
((e,d),(c,(a,b)))



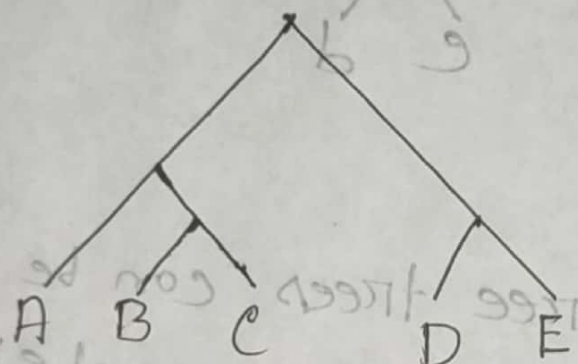
these three trees can be ~~given~~ ~~denoted~~ shown by some rooted tree, ~~given~~ given by ((((a, b), c), (d, e))

* Newick notation can represent
  → rooted binary tree
  → " non binary "
  → unrooted tree

☐ rooted tree for the ∮ newick string
((u1, w2), (v1, w2), (w1, w2))



Rooted tree for ((A, (B, C)), (D, E))



Unrooted tree:

An unrooted phylogenetic tree is a type of phylogenetic tree, that only describes the relatedness of a group of organisms without making assumptions about ancestry.

→ can be generated from rooted by simply omitting the root.

→ if the root of T has two children, then these two children are made adjacent to each other.

* unrooted → rooted (one to many)
* rooted → unrooted (one to one)

NB

* each rooted tree has an unique unroot edge version

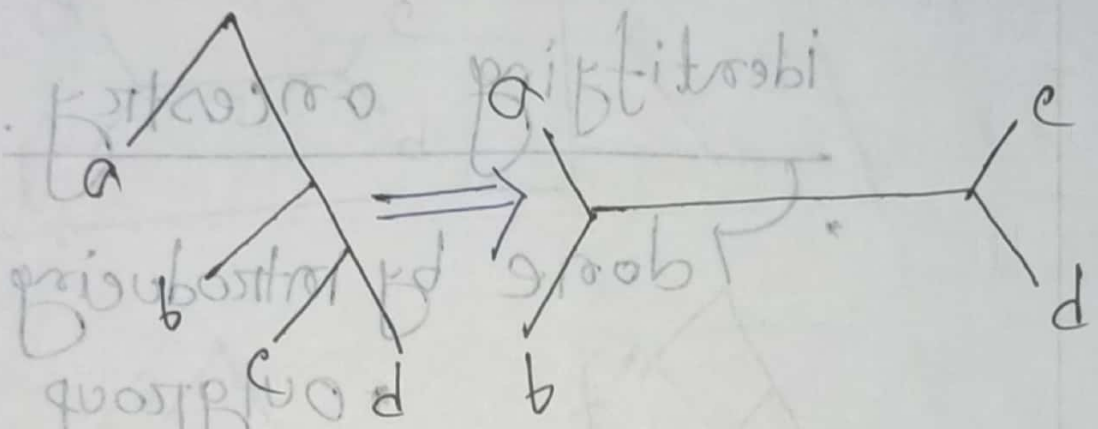
* each unrooted tree has multiple rooted version.

Outgroup:

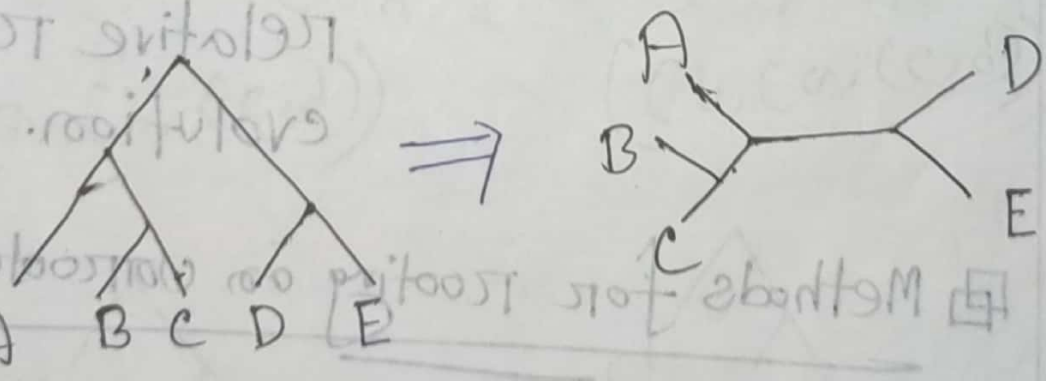
Taxa that are clearly not or as closely related to the rest of the input taxa or the remaining taxa are to each other.

Unrooted tree of rooted tree

(rooted → unrooted)



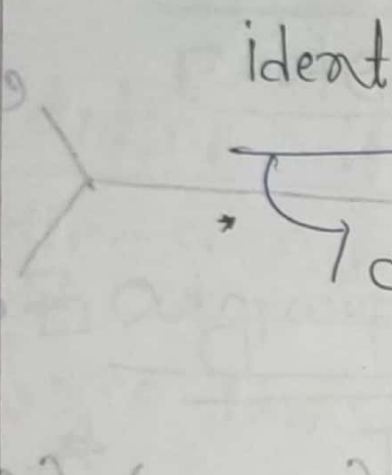
((a,(b,(c,d)))) ((a,b),(c,d))

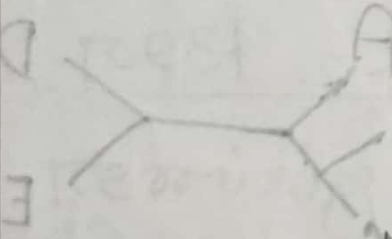


((A,(B,C)),(D,E))

Unrooted tree ⇒ rooted tree

* requires some means of identifying ancestry.
  * done by introducing
    * outgroup
    * additional assumption about the relative rates of evolution.

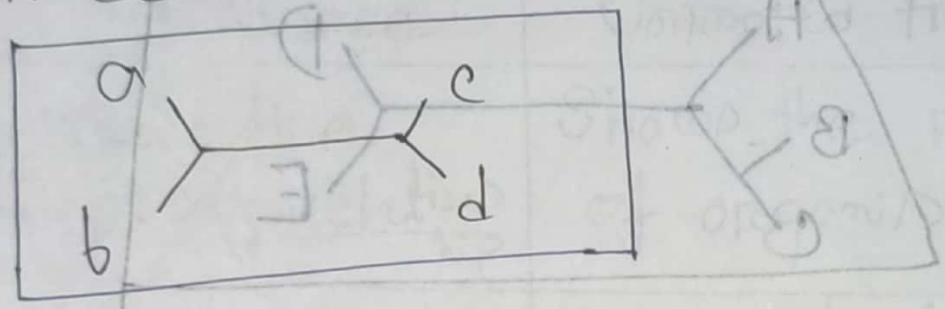


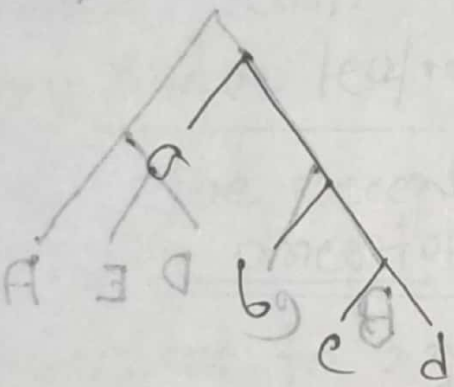


Methods for rooting an unrooted tree:

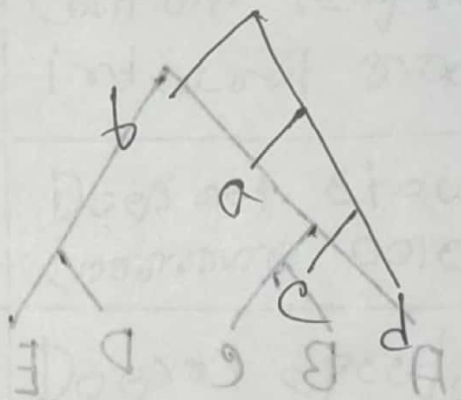
- outgroup rooting
- midpoint rooting
- molecular clock rooting
- Bayesian molecular clock rooting

Rooted version of following unrooted tree:

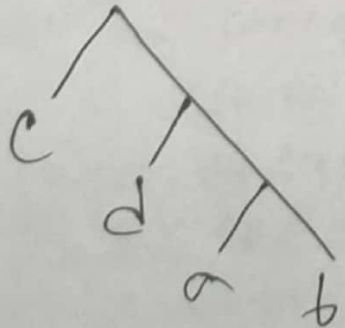




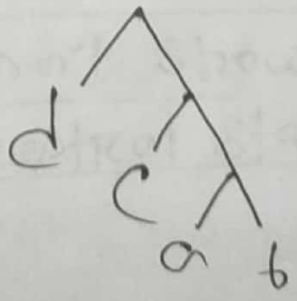
((a, (b, (c, d))



(b, (a, (c, d)))

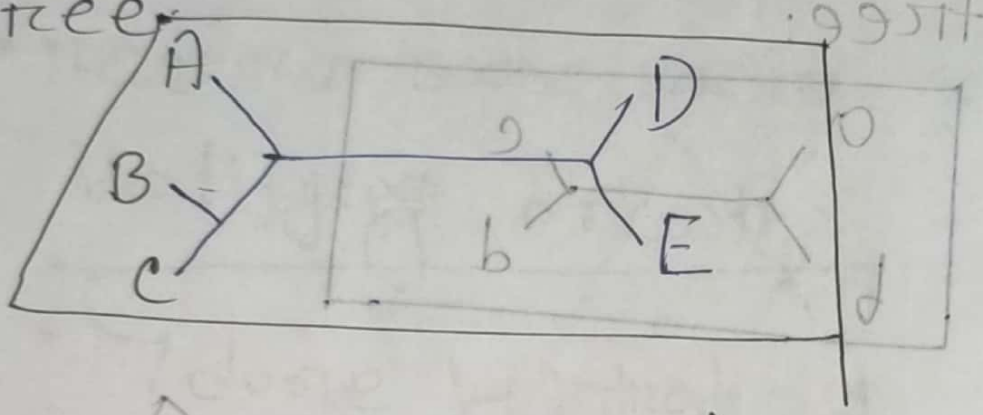


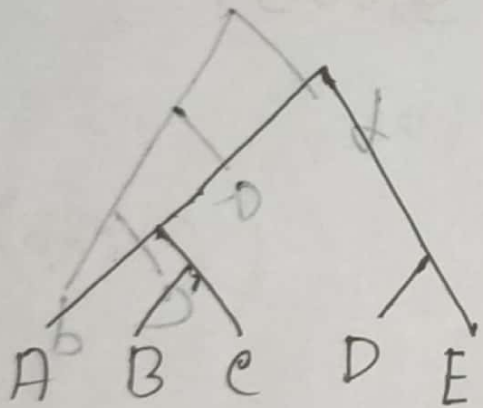
(c, (d (a, b)))



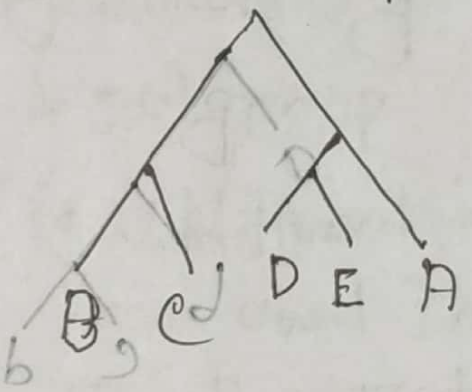
(d, (c, (a, b)))

Rooted tree of following unrooted tree

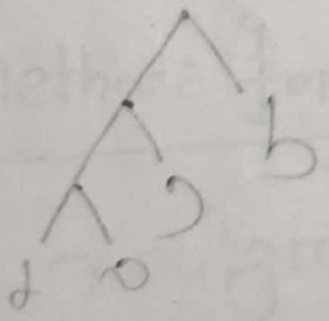




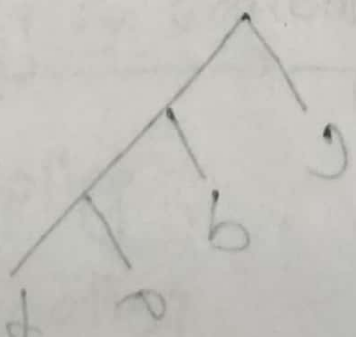
((b,c),a),d))



((b,c),d),a))



((d,a),c),b)



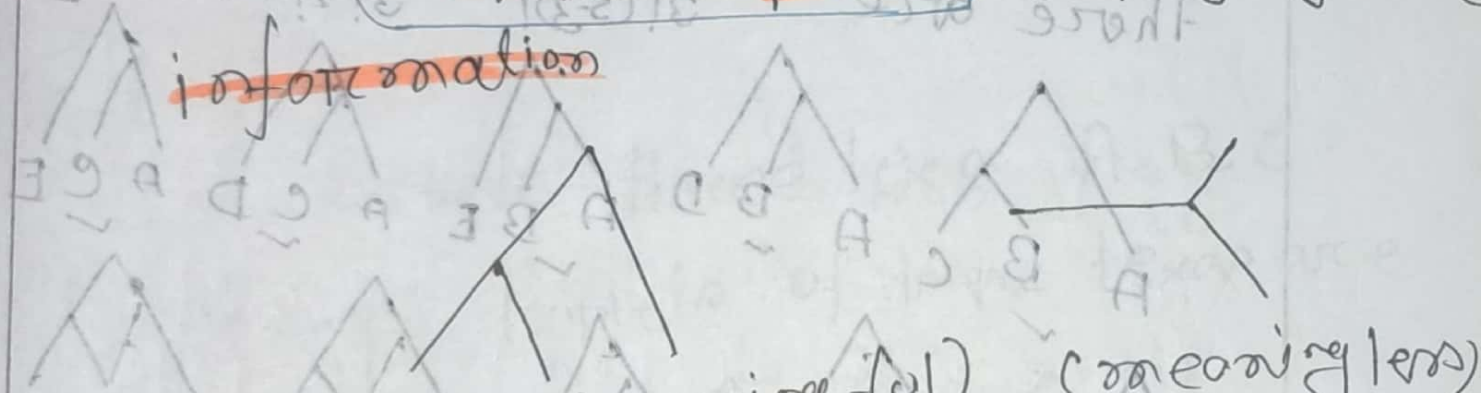
((d,c,b),c)

Rooted tree Vs Unrooted tree

Rooted tree	Unrooted tree
It shows the ancestry relationship	Shows the relatedness of organisms
Contains a root, internal nodes, leaf node	Contains leaf nodes and internal nodes
Shows the recent common ancestor	Does not show any common ancestor
Each root to a node Shows the evolutionary path	Doesn't specify the evolutionary path
Shows the ancestral state of organisms or gene at the bottom to the terminal branches.	Doesn't show the ancestral state

Triplets: Rooted tree with three leaves (species)

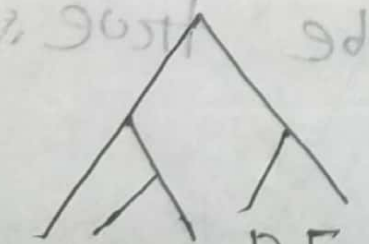
* The most basic piece of phylogenetic information



(meaningful) (meaningless)

* Unrooted triplets are meaningless

Decomposing input gene tree into its induced triplets:



$$\frac{n!}{3!(n-3)!}$$

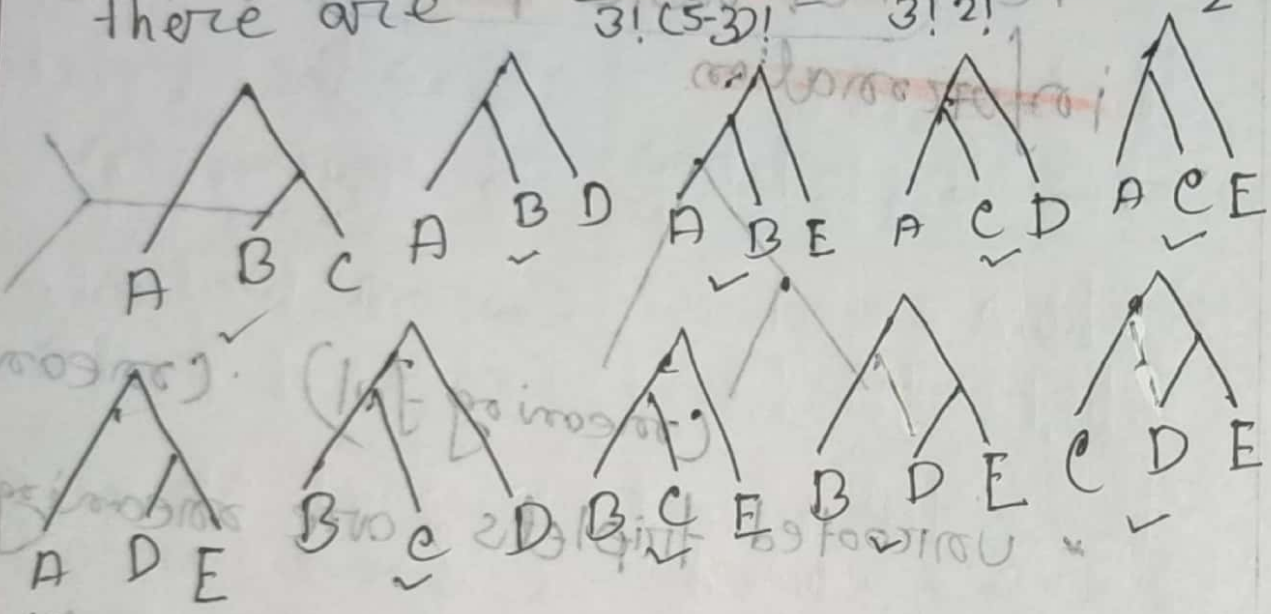
NB:

* Number of induced gene tree (triplet) $= {}^{n}C_3$

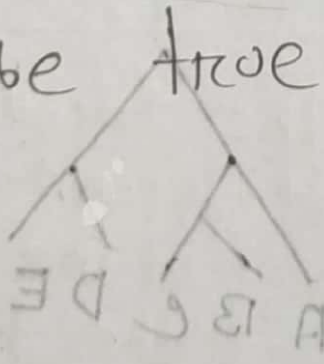
#texCS

ABC | ACD | ADE | BCD | BDE | CDE
ABD | ACE | | BCE | |
ABE | | | | |

There are $\frac{5!}{3!(5-3)!} = \frac{5!}{3!2!} = \frac{4 \times 5}{2} = 10$ triplets



NB: Triptet which come maximum time that will be true spelling (TST)



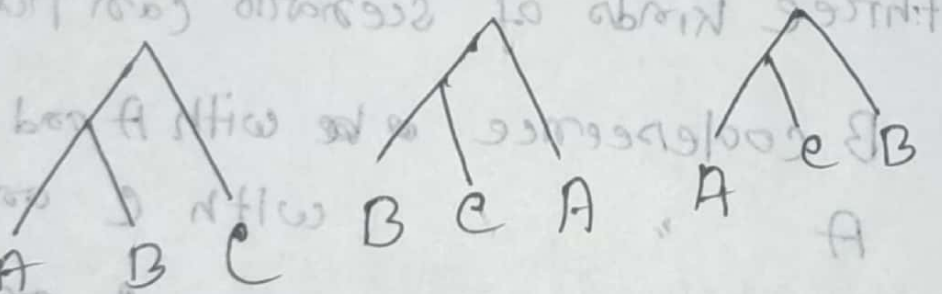
* Triplets and quartets are statistically consistent estimates.

Triplets are statistically consistent estimator of true species tree.

Proof:

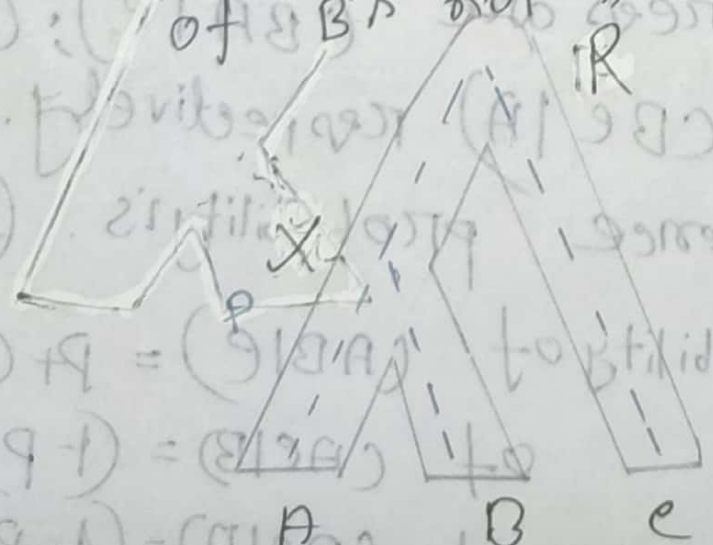
Let, we have three taxa A, B, C.

∴ Induced triplets of these taxa are



Probability of B's coalescence with A is $P_{node\ X}$

∴ " of B's not " with A is $1-P_{node\ X}$



~~For second case B:~~

If B coalescence with A at node X then gene tree is (A,B,C)

If B doesn't coalescence with A at node X then ~~it coalescence~~ three kinds of scenario can happen.

B coalescence with A and then C (A,B,C)

A " with C and then B (A,C,B)

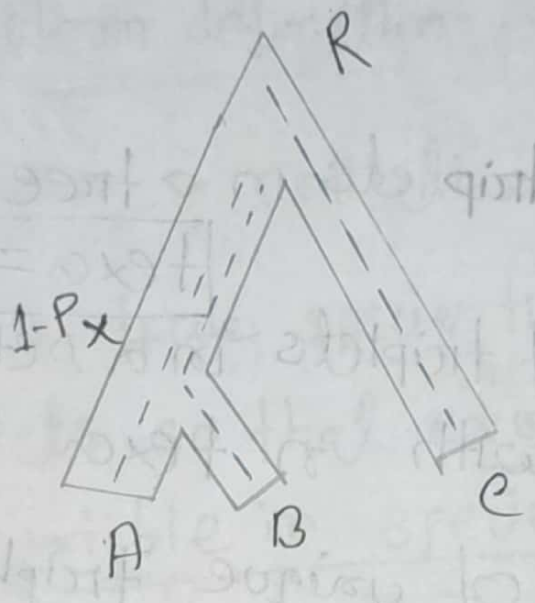
B " with C and then A (B,C,A)

gene trees are (B,A,C); (A,C,B) and (B,C,A) respectively. and their coalescence probability is $(1-P)/3$

∴ probability of (A,B,C) = $P + (1-P)/3$

" of (A,C,B) = $(1-P)/3$

" of (B,C,A) = $(1-P)/3$


R
1-P
x
A
B
C

* Number of triplets in a tree with n texa = $n C_3$

* Number of triplets in a set of K trees with n texa

* Number of unique triplets in a set of K trees with n texa

* Number of unrooted binary tree on n texa = $(2n-5)!!$

* Number of rooted binary tree on n texa = $(2n-3)!!$

Problem defination: ☆☆

Given a collection of gene tree (triplets) find a ~~tree~~ species tree by joining these triplets so that every gene trees compatible in species tree.

Algorithm:

→ If the number of taxa in X is less than 3, just return the tree in X.

Else,

→ Find a sibling pair a, b (which a is always sibling)

→ If no such pair exists return "No tree".

→ Else, continue

→ Remove all rooted triples that ~~induced~~ include a from the set X

→ Recursively compute a tree on the reduced set X' of rooted triple

→ Insert a into the tree by making it sibling to b.

Construct a tree on the leaf set $\{a, b, e, d, e\}$ which is consistent with each of the following triplets. You have to show the intermediate steps of your algorithm.

$ab|e$ $ae|e$

$ad|de$ $be|d$

$ab|d$ $be|e$

$ab|e$ $bd|e$

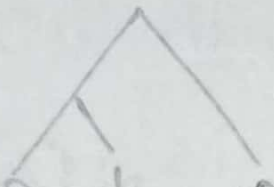
$ae|d$ $cd|e$

Soln

Step 1:

pair of siblings whie are always grouped together are, bc, de,

Choose bc



Step 2

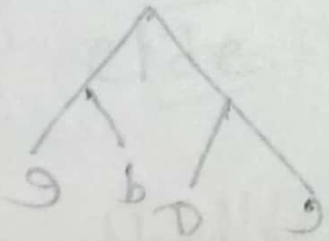
Remove triplets which has b and remaining triplets are

a|de

a c|d

a e|e

c|de



Step-3

pair of siblings which are always grouped together are a e, d e and

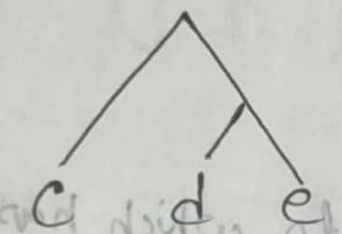
Choose ae

Step-4

Remove triplets with a and

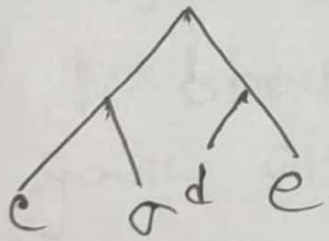
remaining triplets are c|de

Step-5



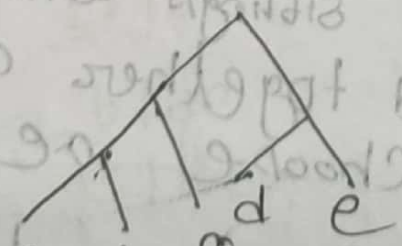
Step-6

Now make a the sister of e.



Step-7

Now make b the sister of e



∴ final tree is cba|de
or ((c,b),a),(d,e))

Construct a tree on leaf tree {a,b,c,d,e}
which is consistent with each of the
following triplets. You have to
show the intermediate steps of your
algorithm.

a|be	a|de
a|bd	b|ed
a|be	b|ce
a|ed	b|de
a|ce	c|de

Sol:

Step-1: Sibling that is always grouped
together is de.

Step2:
Remove triplets with d and
remaining triplets are

a|bc b|cc
a|be
a|ce

Step 3

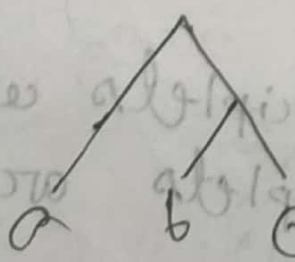
Sibling that is always grouped together is ce

Step 4

Remove triplets with ce and remaining triplets are

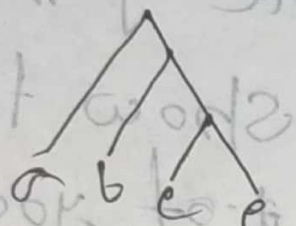
a|be

Step 5



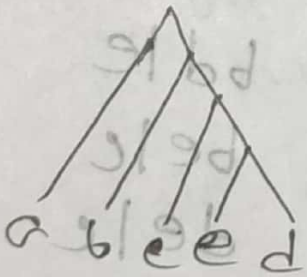
Step-6

make c as the sister of e



Step-7

~~Make~~ Make d as the sister of e



Construct a tree on the leaf set {a, b, c, d, e} which is consistent with each of the following triplets. You have to show the intermediate steps of your algorithm.

ac|b	ac|e
bd|a	bd|e
be|a	bd|e
de|a	be|c
ac|d	de|c

Soln:

Step 1 Siblings that are always grouped are ac, bd.

Choose ac.

Step-2

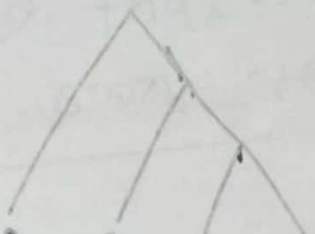
Remove triplets with a and remaining a

bdlc

bdle

bele

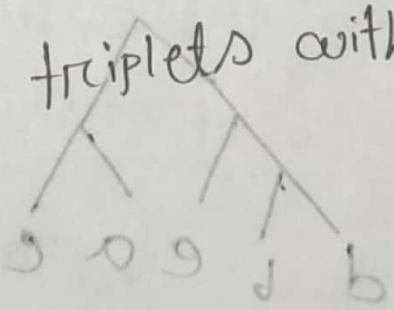
dele.



Step-3

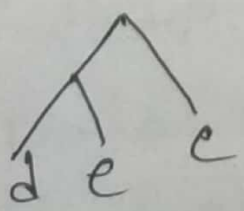
Sibling that is always grouped together is bd.

Step-4 : Remove triplets with b and remaining are.



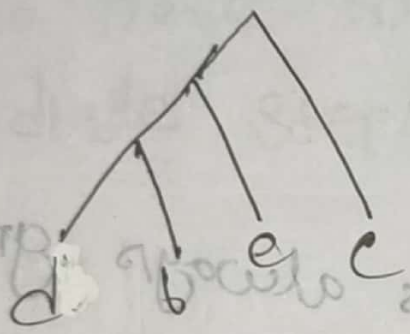
dele

Step-5



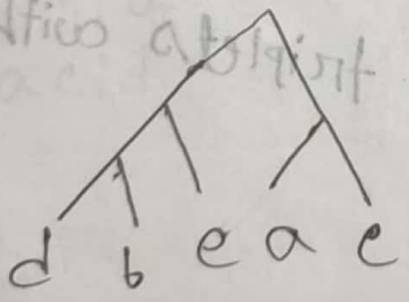
Step-6

Make b as the sibling ~~of~~/sister of d



Step-7

Make a as the sister ~~of e~~ of c.



→ group of organisms that share common ancestor

Clades

L(T) is called the leafset of T.

T_v denotes the subtree of T below node v. (The clades of T are the subset of L(T) that are equal to the $L(T_v)$ for some vertex v.)

Cluster:

Set of taxa in a subtree

Type of clades:

trivial clades:

non trivial clades

trivial clades: Let T be a rooted tree on leafset S.

All the singleton clades and the set S are trivial clades.

non-trivial clades:

All the clades that are not trivial.

Properties of clade to become Compatible

- contain next:
- disjoint
- One clade's leaf set fully contain the leaf set of another clade
- clade that doesn't contain only singleton clades

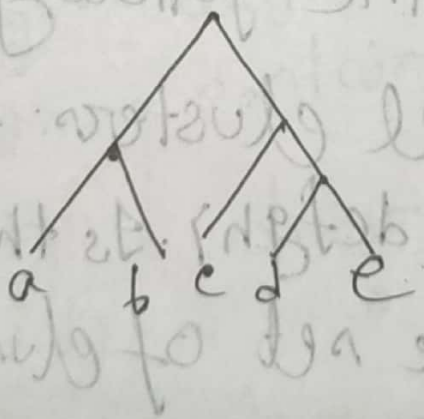
trivial clades and non-trivial clades of tree

$T = ((a,b),(c,(d,e)))$.

trivial clades are $\{a\}, \{b\}, \{c\}, \{d\}, \{e\}, \{a,b,c,d,e\}$

non-trivial clades are $\{a,b\}; \{c,d,e\}; \{d,e\}$

$\therefore$ Clades $(T) = \{\{a\}, \{b\}, \{c\}, \{d\}, \{e\}, \{a,b,c,d,e\}, \{a,b\}, \{d,e\}, \{c,d,e\}\}$



Compatibility:

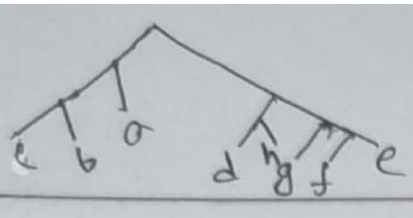
Two clades/clusters are compatible if they can "co-exist" in a binary rooted tree

- A clade can't overlap another clade to become compatible

April - 2021

Consider the following set of non trivial clusters: {bc, ef, dh, abc, efg, defgh}. Is this a compatible set of clusters?

If yes show the corresponding tree. If not explain why.



Soln

Now find out the relation between every possible pair of non-trivial cluster

be, ef → disjoint
be, dh → "
*be, abe → containment
be, efg → disjoint
be, defgh → "

dh, abe → disjoint
dh, efg → disjoint
*dh, defgh → containment

abe, efg → disjoint
abe, defgh → "

ef, dh → disjoint
ef, abe → "
ef, efg → containment
ef, defgh → "

efg, defgh → containment

So there is no overlapping relation between any pair of non-trivial cluster.

So given set is compatible.

April 2020

Consider the following set of non-trivial clusters {be, fg, abc, dfg, edfg, bedefg}. Is this a compatible set of clusters? If yes, show the corresponding tree. If not why.

Sol^n:

Now find out the relation between every possible pair of non-trivial clusters.

be, fg → disjoint
be, abc → containment
be, dfg → disjoint
be, edfg → "
be, bedefg → containment

fg, abc → disjoint
fg, dfg → containment
fg, edfg → "
fg, bedefg → "

abe, dfg → disjoint
abe, edfg → "
abe, bedefg → overlap

dfg, edfg → containment
dfg, bedefg → "

edfg, bedefg → containment

As there is a overlapping relationship so given set is not compatible.

Bipartition:

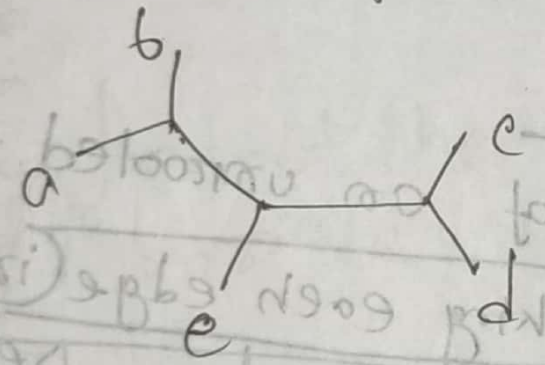
Bipartation of an unrooted tree formed by taking each edge in turn and writing down the two sets of leaves that would be formed by deleting the edge.

$\pi(e)$ used to denote bipartition by edge e, with $\pi(e) = A|B$ where A is one half of the bipartition and B is other

two types of bipartitions:

1. trivial bipartition
2. non-trivial bipartition

Find out the trivial and non-trivial bipartition of tree



Sol^m

non-trivial bipartitions:

$\{a, b\} | \{c, d, e\}$

~~$\{c, d\}$~~

$\{a, b, e\} | \{c, d\}$

Trivial bipartitions:

$\{a\} \mid \{b, c, d, e\}$

$\{b\} \mid \{a, c, d, e\}$

$\{c\} \mid \{a, b, d, e\}$

$\{d\} \mid \{a, b, c, e\}$

$\{e\} \mid \{a, b, c, d\}$

NB:

Bootstrapping:

This is a statistical method for calculating the sampling distribution of an estimator by sampling with replacement from original sample.

Anomalous gene tree (AGT)

~~Most probable gene trees with a topology~~ Gene trees with a topology different from species tree that are more probable to observe than congruent gene tree.

NB: No anomaly zone in 3 species triplets.

Bootstraping:

It is a statical method for calculating the sampling distribution of an estimator by sampling with replacement from original sample.

It is any test or metric that uses random sampling with replacement and falls under the ~~bread~~ broader class of resampling

NB: Branch length in a phylogenetic tree indicate
- confidence
- mutation rate
- time

Estimating branch support by using bootstraping method

→ input sequence alignment is used to generate a large number bootstrap replicate dataset

→ bootstrap replicate is data matrix with the same dimensions of original matrix

→ column of bootstrap replicate is obtained by sampling with replacement → from the original data

- suppose a original input has
  - 4 species
  - sequence length = 200nt
- each time a column is chosen randomly with replacement.
- After 200 times replacement

A new input set is generated

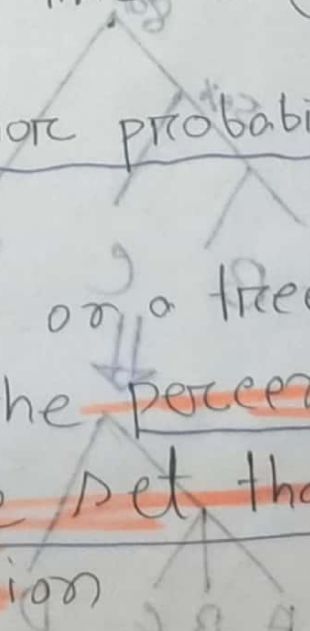
* 100 or more input data is generated
* A phylogeny is estimated on each bootstrap dataset and original dataset using same method

Calculating branch support

* Characterize the support for edge in the tree T.
* An edge e in T has bipartation A|A'
* From bootstrap tree T determine the % of tree for each edge e of same bipartition like original tree T.

☆ branch support/posterior probabilities:

It is defined Branch support on a tree is are obtained by using the percentage of the trees in the set that include the same bipartition

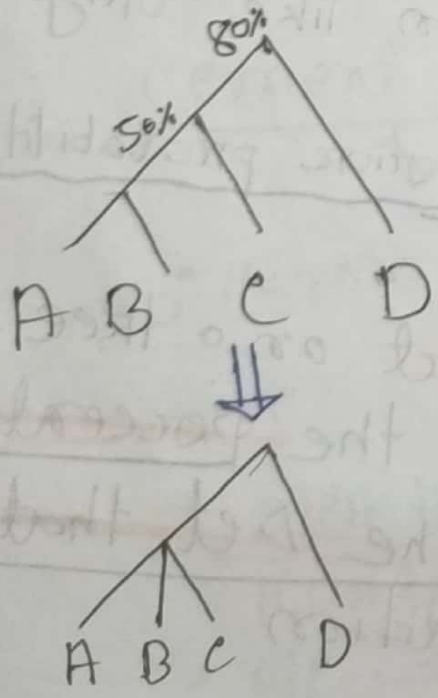


NB

High bootstrap support can be misleading when the ~~estimal~~ estimation method is not statically consistent.

* support value < 50% : unreliable
* support value > 95% : reliable
* ~~Sup~~ support value (50% – 95%): opinions differ as to the reliability of edges.

It may generate non-binary tree



{AB, eD} → 50%

{ABC, D} → 80%.

tree construction methods:

- distance based method
  - * UPGMA
  - * Neighbor Joining
- parsimony based method
  - * Fitch's algorithm
  - * Sankoff's algorithm
- maximum likelihood

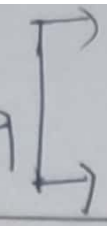
Distance based estimation

2 step processes

1. Computing distance matrix
2. Computing a tree from a distance matrix

two algorithms

1. UPGMA → produce a rooted tree
2. NJ → " an unrooted "



UPGMA (Unweighted pair group method with arithmetic mean)

→ finds a pair (x,y) that have the smallest distance and make them siblings

→ these two are then replaced by the cluster {x,y}

→ the distance from {x,y} to every other taxon z is defined to be the average of d(x,z) and d(y,z)

$$d(\{x,y\},z) = \frac{d(x,z)+d(y,z)}{2}$$

→ the process is repeated until all the taxa are merged into a single cluster

Requirement: ☆☆

→ Ultra metric matrix

→ Additive

* UPGMA fail when the distance does not obey strict molecular clock (not ultrametric)

d(root→leaf) are same.

Ultrametric matrix

An n×n matrix M corresponding to distances between the leaves in a rooted edge-weighted tree T where the sum of the edge weights in the path from the root to any leaf of T does not depend on the selected leaf.

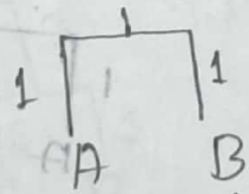
~~or Sum~~

Constructing tree from following distance matrix

	A	B	C	D
A	0	2	16	16
B		0	16	16
C			0	10
D				0

Step-1: As d(A,B) is smallest make a cluster {A,B}

d(A,B) = 2



Step-2

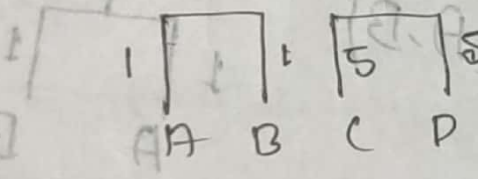
Replace A,B with cluster AB.

$$\therefore d(AB,C) = \frac{d(A,C) + d(B,C)}{2} = \frac{16+16}{2} = 16$$

$$d(AB,D) = \frac{d(A,D) + d(B,D)}{2} = \frac{16+16}{2} = 16$$

	AB	C	D
AB		16	16
C			10
D			

Step-3 : Choose C,D and as d(C,D) is smallest and make cluster {C,D}



Step-4

- Replace C,D with {C,D}.

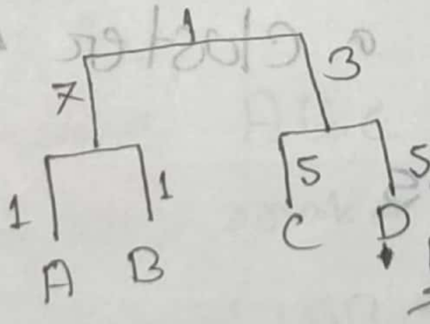
$$\therefore d(AB,CD) = \frac{d(AB,C) + d(AB,D)}{2}$$

$$= \frac{16+16}{2} = 16$$

	AB	CD
AB		16
CD		

Step-5 Choose AB, CD and make a Cluster ABCD as there is only one ~~dis~~ possible Cluster.

$d(AB, CD) = 16$

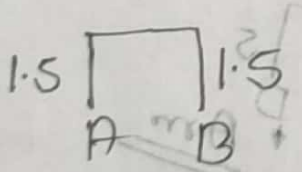


Ans

Constructing tree from following distance matrix

	A	B	C	D	E
A	0	3	7	10	10
B		0	7	10	10
C			0	10	10
D				0	8
E					0

Step-1 : Choose A, B as d(A,B) smallest and make a cluster {A, B}



Step-2: Replace d A, B with cluster {A, B}

$$d(AB, C) = \frac{d(A,C) + d(B,C)}{2}$$

$$= \frac{7+7}{2} = 7$$

$d(AB, D) = \frac{d(A,D) + d(B,D)}{2} = \frac{10 + 10}{2}$

$= 10$

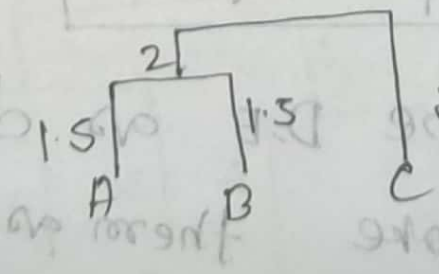
$d(AB, E) = \frac{d(A,E) + d(B,E)}{2} = \frac{10 + 10}{2}$

$= 10$

	AB	C	D	E
AB	0	7	10	10
C		0	10	10
D			0	8
E				0

Step-3 choose AB, C or d(AB, C) is smallest and make ABC or a cluster

$d(AB, C) = 7$



Step-4 Replace AB, C with ABC

$$d(ABC, D) = \frac{d(AB, D) + d(C, D)}{2}$$

$$= \frac{10 + 10}{2}$$

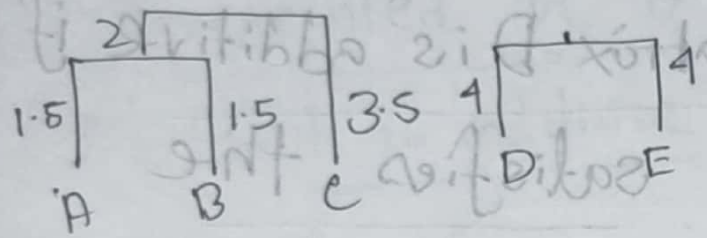
$$= 10$$

$$d(ABC, E) = \frac{d(AB, E) + d(C, E)}{2}$$

$$= \frac{10 + 10}{2} = 10$$

	ABC	D	E
ABC		10	10
D			8
E			

Step-5 Choose D, E as d(D, E) smallest and make them as a cluster.



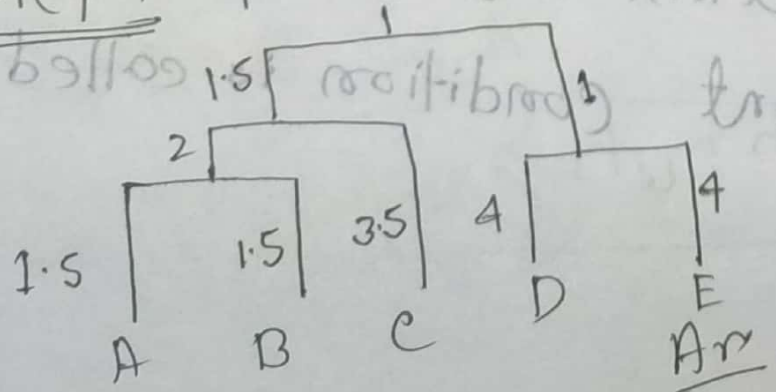
Step-6 Replace D, E with DE

$$d(ABe, DE) = \frac{d(ABe, D) + d(ABe, E)}{2}$$

$$= \frac{10 + 10}{2} = 10$$

	ABe	DE
ABe		10
DE		

Step-7 make ABe, DE as a cluster.



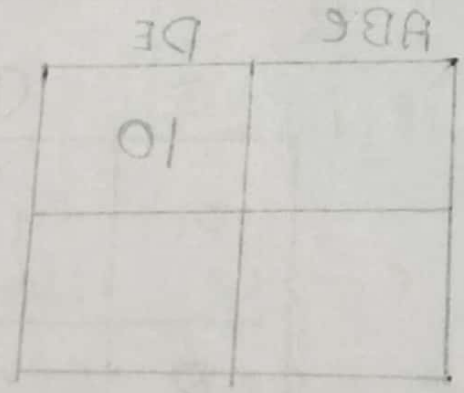
Four point theorem

A $n \times n$ matrix D is additive if and only if it satisfies the "four point condition" for all indices i, j, k, l which is that the median and largest of the following three values are the same.

$D_{i,j} + D_{k,l}$

$D_{i,k} + D_{j,l}$

$D_{i,l} + D_{j,k}$.



Additive: A matrix that satisfies the four point condition is called additive

theorem: A distance matrix can be represented by a tree if and only if it is additive

theorem:

Ultrametric: A dissimilarity map δ on X is an ultrametric if for every three distance $x, y, z \in X$

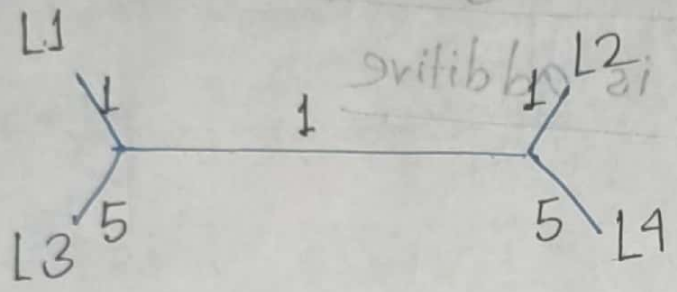
$$\delta(x,y) \leq \max\{\delta(x,z), \delta(y,z)\}$$

triangle inequality

An $n \times n$ matrix d is said to satisfy the triangle inequality if for all i, j, k,

$$d_{ik} \leq d_{ij} + d_{kj}$$

Example of additive matrix



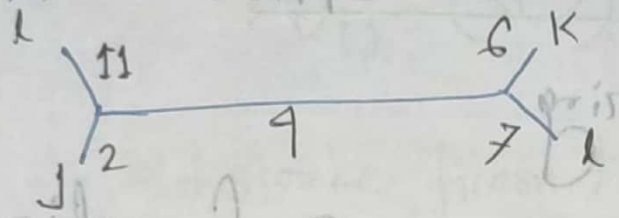
$x = D_{(L1,L3)} + D_{(L2,L4)} = (1+5) + (1+5)$
$= 6+6 = 12$

$y = D_{(L1+L2)} + D_{(L3,L4)} = (1+1+1) + (5+1+5)$
$= 3+11 = 14$

$z = D_{(L1,L4)} + D_{(L3,L2)} = (1+1+5) + (5+1+1)$
$= 7+7 = 14$

As $y = z > x$, so, it is an additive matrix.

Determine if the following is additive or not



$x = D_{ij} + D_{KL} = (11+2) + (6+7) = 13+13$
$= 26$

$y = D_{iK} + D_{jL} = (11+9+6) + (2+9+7) = 21 + 13$
$= 34$

$z = D_{iL} + D_{jK} = (11+9+7) + (2+6+9) = 22 + 12$
$= 34$

$y = z > x$

So it is an additive matrix

Neighbor Joining (NJ)

→ A distance method with agglomerative clustering

→ Compute the Q function on the given distance matrix

→ Clustering is similar to UPGMA, but computing the distance from a clustered not to to the others is different from UPGMA

Input: $n \times n$ dissimilarity matrix d with $n \geq 4$

Output: Unrooted tree with n leaves labelled $1 \ldots n$

initialization: Compute the $n \times n$ matrix Q, defined by

$$Q_{i,j} = (n-2)d_{ij} - \sum_{k=1}^{n} (d_{ik} + d_{jk})$$

total distance from i to all other

" " " (i,j) to " " "

$D \rightarrow D^*(Q) \rightarrow D'$

recursively apply neighbor joining on D'

(Distance → NJ matrix)

$$D^*_{ij} = (n-2)\, D_{ij} - \text{Total distance}_d(i) - \text{Total distance}_d$$

$$\Delta_{ij} = \frac{\text{Total distance}_d(i) - \text{Total distance}_d(j)}{(n-2)}$$

$$\text{Limb length}\ (i) = \frac{1}{2}(D_{ij} + \Delta_{i,j})$$

$$\text{Limb length}\ (j) = \frac{1}{2}(D_{ij} - \Delta_{i,j})$$

Let cluster $m = \{i,j\}$

$$D_{k,m} = \frac{1}{2}(D_{ki} + D_{kj} - D_{ij})$$ (NJ matrix → Distance)

Neighbor-Joining theorem:

Given an additive matrix D, the smallest element $D^*_{i,j}$ of its neighbor-joining matrix D^* corresponds to a pair of neighboring leaves i and j in Tree(D).

Construcing tree from the following Distance matrix (Neighbor Joining)

	i	j	k	l
i	0	13	21	22
j	13	0	12	13
k	21	12	0	13
l	22	13	13	0

Sol^{n}

Step-1: ~~Distance matrix to Neighbor-Joining matrix~~

Calculate total distance

$TD_i = 56$

$TD_j = 38$

$TD_k = 46$

$TD_l = 48$

Step-2

$D \rightarrow D^*$

$n = 4$

$\therefore (n-2) = 4-2 = 2$

	i	j	k	l
i	0	-68	-60	-60
j	-68	0	-60	-60
k	-60	-60	0	-68
l	-60	-60	-68	0

$D^*_{ij} = 2 \times D_{ij} - TD_i - TD_j$

$= 2 \times 13 - (56+38)$

$= 26 - 94$

$= -68$

$D_{ik} = 2 \times D_{ik} - TD_i - TD_k$

$= (2 \times 21) - 56 - 46$

$= 42 - (102)$

$= -60$

$D_{il} = 2 D_{il} - TD_i - TD_l$

$= (2 \times 22) - (56 + 48)$

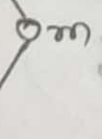
$= 44 - 104$

$= -60$

-68 is smallest value

∴ Construct a cluster

$m = \{i, j\}$



Step-3 $D^* \rightarrow D^-$

	m	k	l
m	0	10	11
k	10	0	13
l	11	13	0

$D_{mk} = \frac{1}{2}(21+12-13)$

$= \frac{1}{2}(21-1)$

$= \frac{1}{2} 20 = 10$

$D_{ml} = \frac{1}{2}(22+13-13)$

$= 11$

Step -4

Calculate TD_m or D'

$TD_m = 21$

$TD_k = 23$

$TD_l = 24$

Step-5

$D' \rightarrow D''$

	m	k	l
m	0	-34	-34
k	-34	0	-34
l	-34	-34	0

Choose (k, l, or m) or Cluster m or = {k, l}

$m = 3$

$r - 2 = (3 - 2) = 1$

$D''_{mk} = 1 \times D'_{mk} - (TD_m + TD_k)$

$= 1 \times 10 - (21 + 23)$

$= 10 - 44$

$= -34$

$D''_{m1} = 1 \times D'_{m1} - (TD_m + TD_l)$

$= (1 \times 11) - (21 + 24)$

$= 11 - 45$

$= -34$

$D''_{kl} = (1 \times 13) - (23 + 24)$

$= 13 - 47$

$= -34$

Step 6

$D^* - D^=$

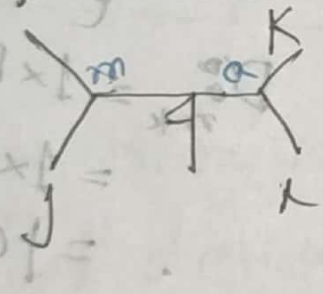
	m	a
m	0	4
a	4	0

$$D_{ma} = \frac{1}{2}(D_{m,k} + D_{m,l} - D_{kl})$$

$$= \frac{1}{2}(10 + 11 - 13)$$

$$= \frac{1}{2}(10 - 2)$$

$$= \frac{1}{2} 8 = 4$$



Calculating limb length

$$\Delta_{ij} = \frac{1}{(n-2)}(TD_i - TD_j)$$

$$= \frac{1}{(4-2)}(56 - 38) = \frac{1}{2} \times 18$$

$$= 9$$

$$\text{Length}(i) = \frac{1}{2}(9 + D_{ij}) = \frac{1}{2}(13 + 9)$$

$$= \frac{1}{2} 22$$

$$\therefore (j) = \frac{1}{2}(D_{ij} - 9) = \frac{1}{2}(13 - 9) = \frac{1}{2} 4 = 2$$

$$= 11$$

Length(K) $= \frac{1}{2}(D_{KL} + \Delta_{KL})$

$= \frac{1}{2}(13 - 1)$

$= \frac{1}{2}\,12 = 6$

Length(L) $= \frac{1}{2}(D_{KL} \pm \Delta_{KL})$

$= \frac{1}{2}(13 - (-1))$

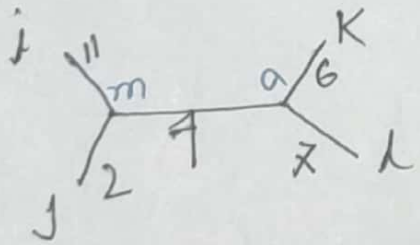
$= \frac{1}{2}\,14 = 7$

$\Delta_{KL} = \frac{1}{(n-2)}(TD_K - TD_L)$

$= \frac{1}{3-2}(23 - 24)$

$= -1$

∴ final tree



Gene tree estimation

- maximum likelihood
- maximum parsimony

(Sequence → tree)

* given sequence
* apply multiple sequence alignment
* run **Maximum likelihood** method

NB
Disad: gene tree estimation error may cause gene tree discordance.

Species tree estimation:

* → combined analysis
  → summary methods

Combined analysis:

* ~~Combining~~
Supergene: Combining all genes to create supergene

→ create supergene(g*)
→ apply sequence based estimation (maximum likelihood, parsimony)

Dis.ad.
→ ignores gene tree discordance

Adv:
→ provide high accuracy.

Summary method:

- Create different gene tree from different gene
- Apply different technique to merge gene trees
- generate a species tree

merging technique

→ maximum voting

→ maximum parsimony

Disadvantages

→ discordance

→ anomaly zone

Gene tree parsimony / maximum parsimony

Find the tree that requires minimum number of changes to explain the data.

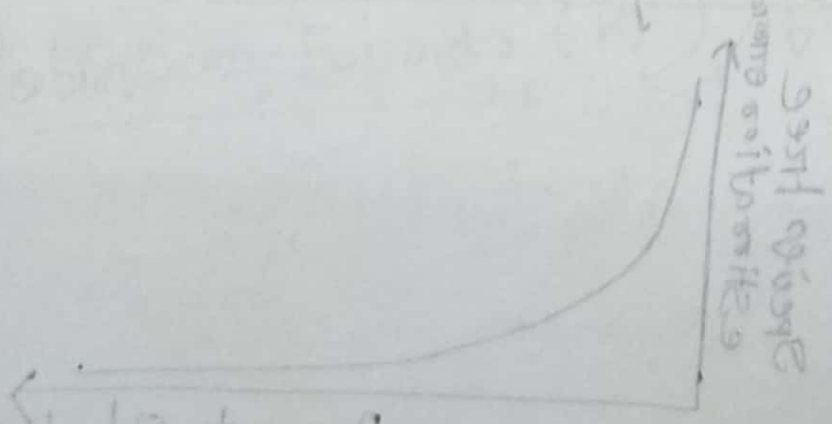
input: A set of rooted binary gene tree with each species having a single copy of gene

output: A species tree that optimizes a particular cost function.

* minimize cost w.r.t duplication
* maximize extra lineage

Parsimony approaches usually have two different components

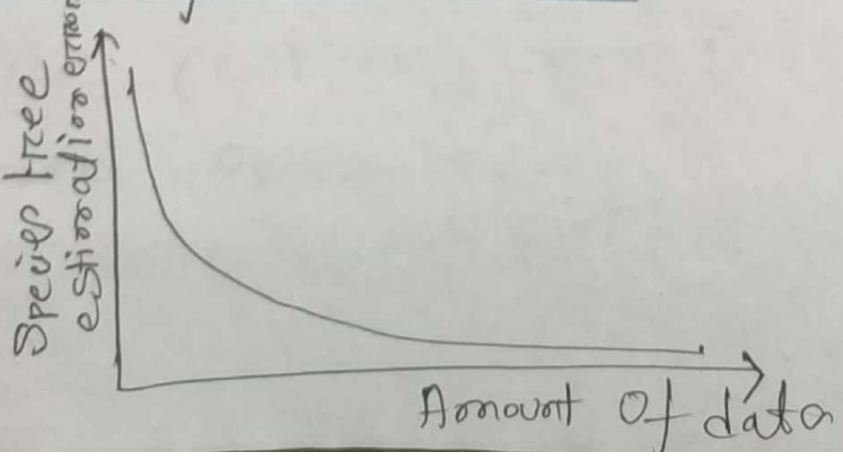
1. Search through many different tree topologies to find a good one
2. Find the minimum number of changes needed to explain the data for a given tree topology.



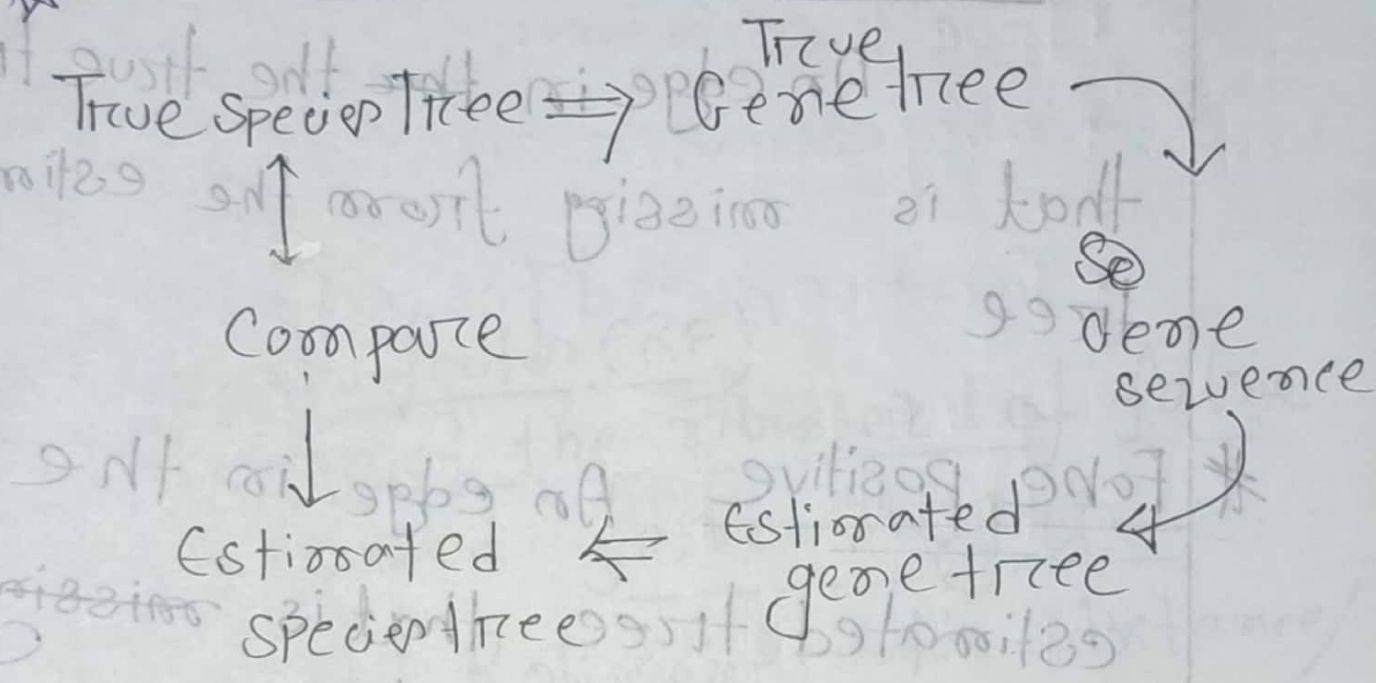
Statistical consistency: (Species tree estimation)

A species tree estimation method is called statistically consistent if the probability of returning true species tree converges to one as the amount of data increases

* we usually assume both the number of sites/ loci and the number of loci increase.



Evaluation procedure:



Metrics

→ False negative (FN) → false negative rate

→ False positive (FP) → false positive rate

→ Robinson-Foulds (RF) / bipartition distance

False negative

An edge in ~~the~~ the true tree that is missing from the estimated tree

False positive An edge in the estimated tree that is ~~missing~~ not in the true tree

false negative rate :

Proportion of the missing edges

$$\frac{\#FN \text{ edges}}{\#\text{ internal edges in the true tree}}$$

false positive rate:

Robinson-Foulds (RF)

Sum of the number of false positive and false negative.

Also called Robinson-Foulds distance / Bipartition distance.

$$RF = \frac{\#FP + \#FN}{2n - 6}$$

n → number of leaves

* true tree is binary (assumed)
* many estimated tree are not binary
* To is binary and T is not binary
false positive error rate will be less than false negative error rate
~~* To is not bi~~
* To and T both is not binary
false negative error rate could be smaller than the false positive error rate.
* So Robinson-Foulds is not equal to the average of fall negative and false positive
* if T is not binary we have to report both FP and FN instead of RF.

Nov-2020/April-2020

Suppose you are trying to construct a species tree on 10 different species. You have sampled 250 genes from each of these 10 species. Your supervisor has asked you to use a method called GTest for constructing trees from sequence alignments, and SPest (which is summary method) for estimating species trees from gene trees.

i) How many times you need to run GTest and SPest to estimate a species tree by summarizing gene trees.

ii) How many times do you need to run GTest and SPest to estimate a species tree by "combined Analysis"?

~~As~~

Ans:

i) Species tree by summarizing gene trees:

GT-est: 250 times

SP-est: 1 time

ii) Species tree by "Combined Analysis"

GT-est: 1 time

SP-est: 0 time

November-2021 / April 2021

Suppose you are trying to construct a species tree on 15 different species. You have sampled 300 gene sequences from each of these 15 species. Your supervisor has asked you to use a method called GTest for constructing trees from sequence alignment and SpTest (which is a summary method) for estimating species trees from gene trees.

i) How many times you need to run GTest and SpTest to estimate species tree by summarizing gene trees?

ii) How many times you need to run GTest and SpTest to estimate species tree if you use "Naive Binning" with 10 bins?

iii) How many times dog you need to run GT-est and SP-est to estimate species tree if you use (statical Binning) assuming that SB returns 12 bins?

iv) How many times do you need to run GTest and SP est to estimate a species tree if you use weighted statical Binning (WSB) assuming that WSB returns 12 bins?

v) How many times do you need to run GTest and SP-est to estimate a species trees using combined analysis?

Ans:

i) Species tree by summarizing gene trees:

GT-est: 300 times

SP-est: 1 time

ii) Species tree by using combined analysis:

GT-est: 1 time

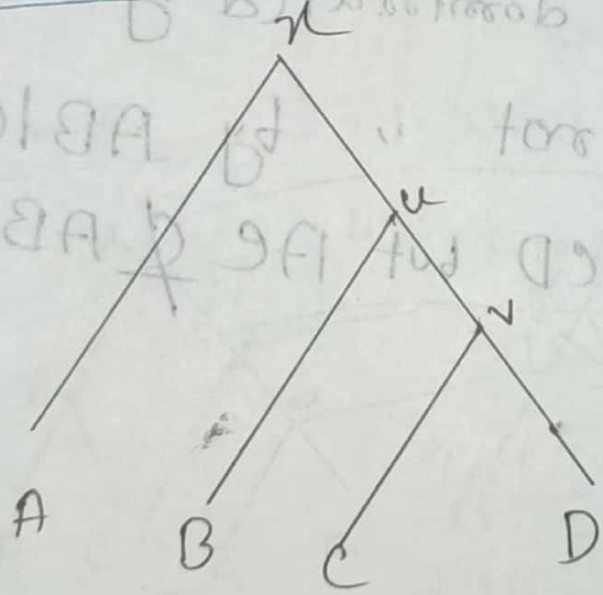
SP-est: 0 time

Subtree-bipartition:

For an internal node u in a binary rooted tree T,

$$SBP(u) = cluster(T_L) \mid cluster(T_R)$$

Example: Subtree bipartition of following tree



for node u B|CD

" " v C|D

" " r A|BCD

Domination

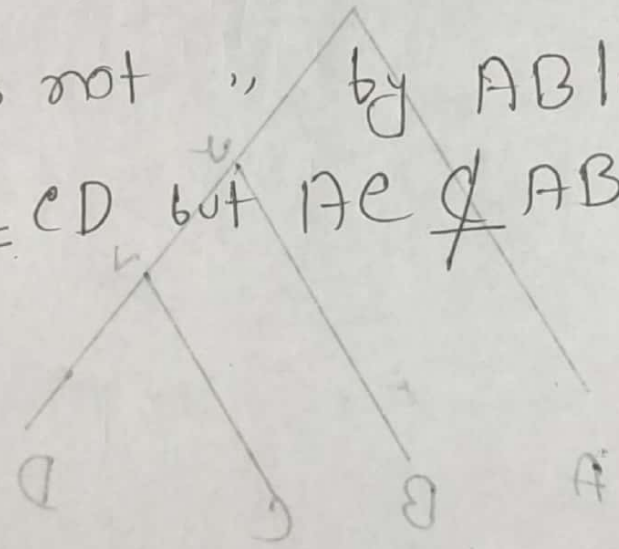
X|Y is dominated by P|Q

if $X \subseteq P$ and $Y \subseteq Q$

Example

* A|CD is dominated by AB|CD
* AE|D is not " by AB|CD

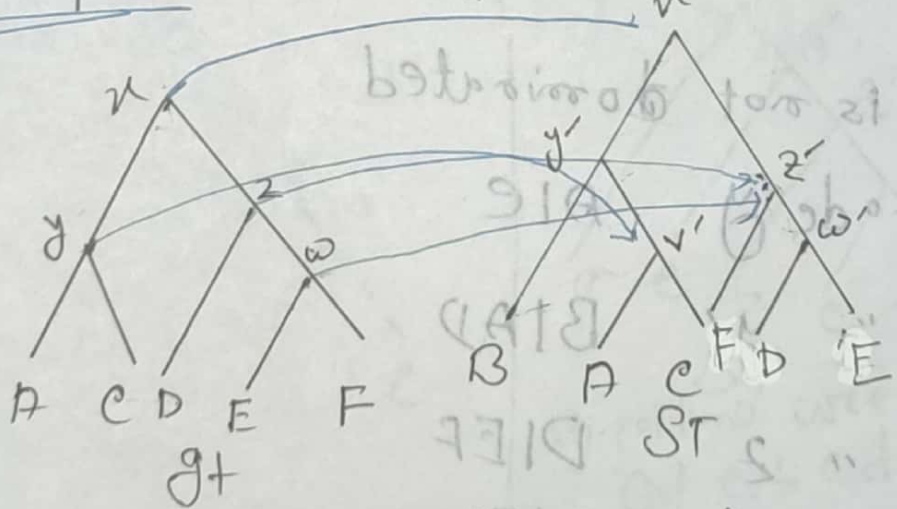
as $D \subseteq CD$ but $AE \not\subseteq AB$



Duplication (alternative dfn)

An internal node of gt is a speciation node if it is dominated by some subtree-bipartition in ST. Otherwise, this is a duplication node.

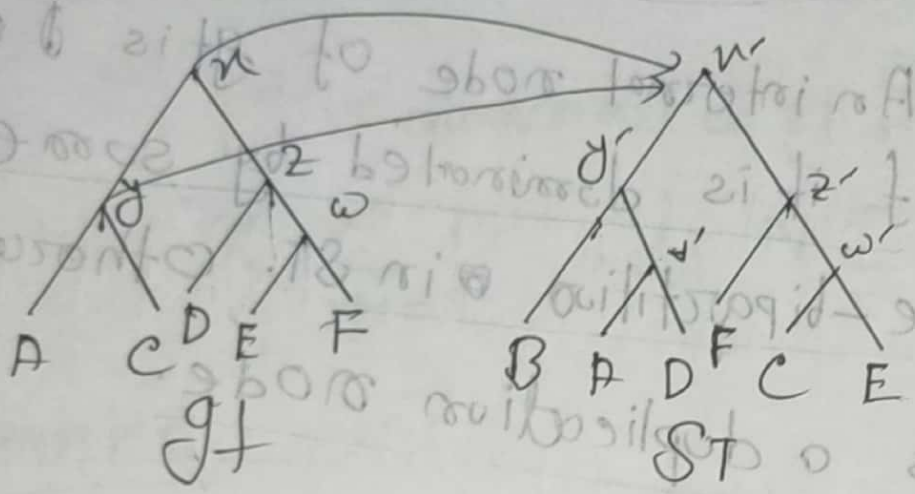
Example



for node x A C | D E F
" " x' A B C | D E F
" " y A | C
" " v' A | C

for node z D | E F
" " z' F | D E

#



for node x AC| DEF

" " x' BAD| EFC

As it is not dominated

for node y A|C

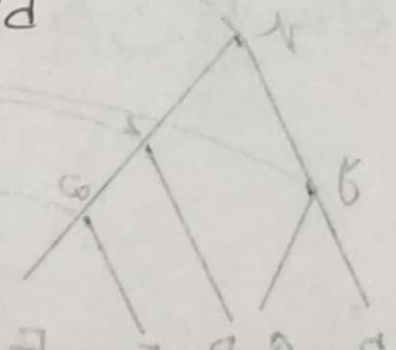
" " y' B|AD

" " z D|EF

" " w E|F

" " z' F|EC

" " w' C|E

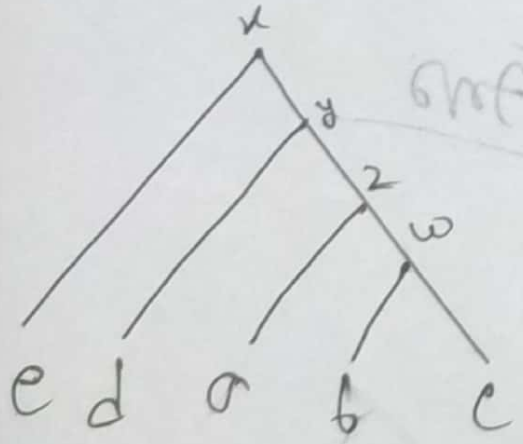


April 2021

Consider the following gt and ST. Find the sets of subtree-bipartitions in gt and ST. Find the subtree-bipartitions in gt that are dominated by some subtree bipartitions in ST.

$gt = (e, (d, (a, (b, c))))$

$ST = (a, (e, (b, (c, d))))$



gt

Bi-partitions are

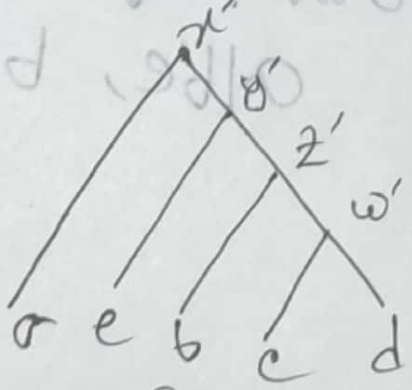
e | d a b c

d | a b c

a | b c

b | c

Ans



ST

Bi-partitions are,

a | e b c d

e | b c d

b | c d

c | d Ans

for a|be and a|ebed

$a \subseteq a$, $be \subseteq ebed$

∴ a|be is dominant

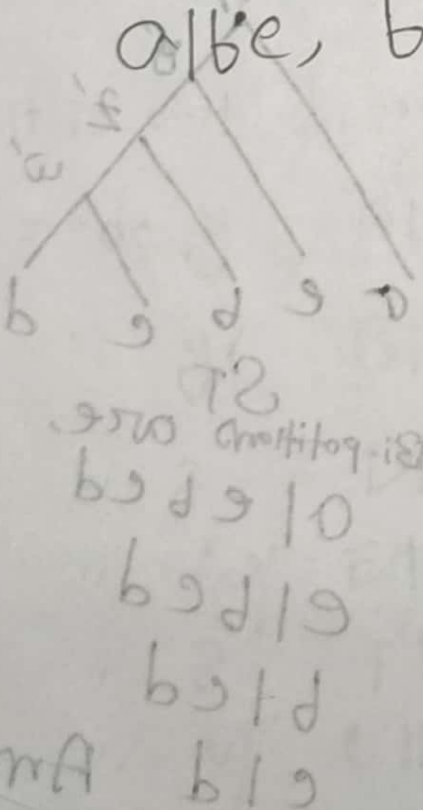
for b|e and b|ed

$b \subseteq b$, $e \subseteq ed$

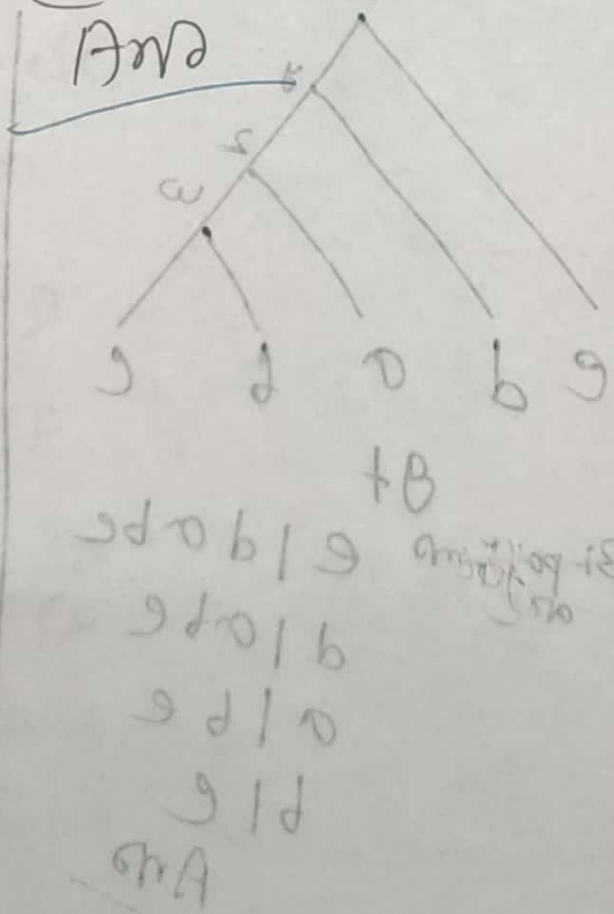
∴ b|e is dominant

∴ Dominant of sub-tree bipartition are

a|be, b|e



And



Multiple Sequence Alignment (MSA)

MSA:

** It reflects evolutionary processes operating on sequences so that the homology can be inferred.

Homology: Two letters in two sequences are homologous if they descend from a letter in a common ancestor.

NB

* MSA → estimating tree
* errors in MSA estimation tend to produce errors in estimated tree
* MSA introduces "-" in the sequences
* "-" sometimes treated as an additional state

* 5 states for nucleotide alignment (RNA, DNA)
* 21 " " protein "
* Column (sites) within the alignment define the homology

Evolutionary Process operating on MSA:

- Substitutions (mutation)
- Insertion and deletion (Indels)
- Rearrangement
  - inversions
  - transpositions
- Duplication of regions

Note: Most alignment methods stretch out sequences so that they line up well and so only address substitutions and indels

Inversion and transposition

Inversion:

S_1 = ACCAGTCACCTA

S_2 = ACCACTGACCTA

Sequence segment are same but they change their direction between two sequence segment of

Transposition:

S_1 = ACCAGTCACCTA

S_2 = ACCCCTAAGTCA

Two segments of sequence swap their location between two sequences.

Example of rearrangement

Turning Cabbage into Turnip

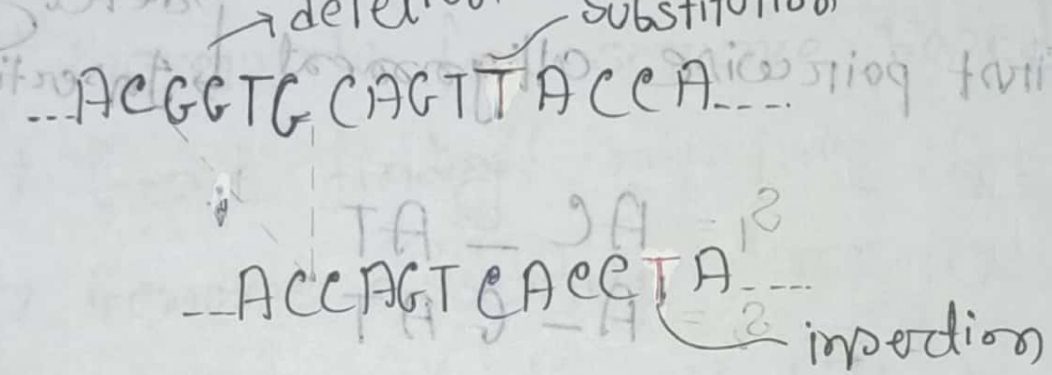
→ Share a common ancestry,
but they look and taste different

→ 99% similarity between genes

→ Differ in gene order

Example:

Evolving one sequence into another sequence through a combination of insertion, deletion and substitution:



Alligonment / Evolving

S_1 = ACGGTGCAGTTACCA--

S_2 = AC---- CAGTC ACCTA --

Example

- Evolving sequence

$S_1 = ACAT$

$S_2 = AGAT$

First pairwise alignment → deleting C
↳ Inserting G

$S_1 \rightarrow$ AC _ AT

$S_2 \rightarrow$ A _ G AT

Second pairwise alignment

A _ C AT

A G _ AT

There is more than one representation as a pairwise alignment, each of which accurately reflects true homology.

Conserved region don't necessarily mean that mutation didn't happen.

Importance of MSA

- Shared ancestry (Homology):
  - Comparative Genomics
- Prerequisite for sequence based phylogeny
- Gene finding
- Genome assembly
- Protein attribute prediction
- Conserved regions, motifs can be identified
  - Highly conserved DNA sequences are thought to have functional value

mutation occurs but it causes death

Lethal Mutation: A type of mutation in which the effects can results in death or reduce significantly the expected longevity of an organism carrying mutation.

Types of mutation:

→ Good
→ Bad
→ Silent

Good: Mutation that enhances the organ's function

Bad: Mutation that cause harmful traits (Huntington disease)

Silent: Mutation that cause no difference in the function of the organism

NB: Lethal dominant genes are rare because the organism dies.

types of sequence alignment: Implemented by DP

→ Local alignment → Smith-waterman

→ Global " → Needleman-wunsch

Local alignment:

match a query with a substring (a portion) of your subject

→ finds regions of similarity in parts of the region

→ searching for local similarities → within large sequence

Global alignment

End-to-end alignment of input sequences

→ Find the best alignment across the whole sequences

→ Comparing two gene → with similar functions

Needleman-Wunsch algorithm
(Global alignment)

For two sequences s and t

$$V_{i,j} = \max \begin{cases} V_{i-1,j} + \delta(s_i, -) \\ V_{i,j-1} + \delta(-, t_j) \\ V_{i-1,j-1} + \delta(s_i, t_j) \end{cases}$$

Alphabet:

DNA: {A, C, G, T}

Scoring scheme:

$\delta(x, x) = 1 \longrightarrow$ match

$\delta(x, -) = -2 \longrightarrow$ indel (insertion/deletion) gap

$\delta(x, y) = -1 \longrightarrow$ mismatch

→ S and t are input sequences

Needleman Wunsch (S,t) pseudo code:

```
m ← length(S)
n ← length(t)
V[0,0] ← 0
for i in 1:m                          → Column initialization
    V[i,0] = V[i-1,0] + δ(s_i, -)
for j in 1:n                          → row initialization
    V[0,1] = V[0,j-1] + δ(-, t_j)

for i in 1:m
    for j in 1:n
        V[i,j] = max { V[i-1,j-1] + δ(s_i, t_j)
                     { V[i-1,j] + δ(s_i, -)
                     { V[i,j-1] + δ(-, t_j)
```

Align and find the aligned sequences of following two sequences S and t with the help of Needleman-Wunsch algorithm.

S = AAAC
t = AGC

$\delta(x,x) = 1$
$\delta(x,-) = -2$
$\delta(x,y) = -1$

		A	G	C
	0	-2	-4	-6
A	-2	1	-1	-3
A	-4	-1	0	-2
A	-6	-3	-2	-1
C	-8	-5	-4	1

∴ Alignment score = −1

~~Alignments~~

Aligned sequence

First one

A G − C

A A A C

Second one

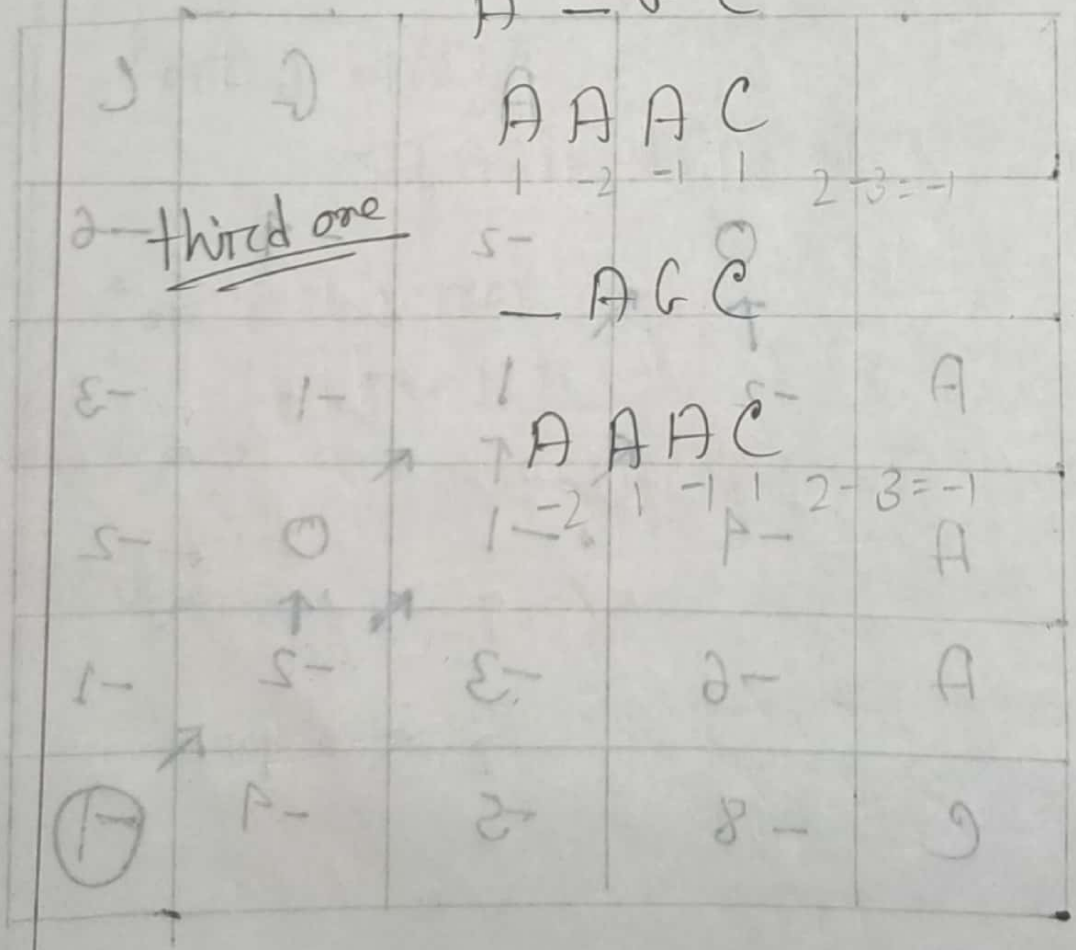
A − G C

A A A C

Third one

− A G C

A A A C



Local alignment:

A min global alignment between substrings of the original sequences

Goal: Find the best local alignment between two strings

Input: Strings s and t as well as a scoring matrix δ

Output: Alignment of substrings of s and t whose alignment score is maximum among all possible alignments of all possible substring of s and t.

*DP to solve the local alignment problem
*Smith Waterman algorithm

Naive approach:

Find the path with maximum score

→ between every pair of vertices (i, j) and (i', j')

* Select the pair with maximum score.

Smith-Waterman:

* find a path with the maximum score from $(0,0)$ to every other vertices.
* add edges of weight 0 in the edit graph.

Recurrence: For two sequences S and t

$$V_{i,j} = \max \begin{cases} V_{i-1,j} + \delta(s_i, -) \\ V_{i,j-1} + \delta(-, t_j) \\ V_{i-1,j-1} + \delta(s_i, t_j) \\ 0 \end{cases}$$

Example

Find the best local alignment of the following two sequences using a gap penalty -2, a ~~mis~~ mismatch penalty -1 and match score = +1.

S_1 = CCG

S_2 = ATCCG

(7 × 5) matrix

$S(s_i, t_j) = S(x, x) = +1$

$S(x, y) = -1$

$S(-, xy) = S(x, -) = -2$

S_2 \ S_1	-	C	C	G
-	0	0	0	0
A	0	0	0	0
T	0	0	0	0
C	0	1	1	0
C	0	1	2	0
G	0	0	0	3

C C G

C C G

Example

Find the best local alignment from the following two sequences by using match score +1 mismatch penalty -1 and gap penalty -2

S1 = TCGGT
S2 = ACCGT

(7×7) matrix

Here, $\delta(x,y) = -1$
$\delta(x,x) = 1$
$\delta(x,-) = -2$
$\delta(-,y) = -2$

S2 \ S1	-	T	C	G	G	T
-	0	0	0	0	0	0
A	0	0	0	0	0	0
C	0	0	1	0	0	0
C	0	0	1	0	0	0
G	0	0	0	2	1	0
T	0	1	0	0	1	2

Alignment-1
C G
C G

Alignment-2
G T
G T

Example

Given that,

DNA, $\Sigma = \{A, C, G, T\}$

Input:

$S = TGTTACGG$

$t = GGTTGACTA$

Scoring scheme:

$\delta(x, y) = -3$

$\delta(x, x) = 3$

$\delta(x, -) = \delta(-, x) = -2$

t \ S	-	T	G	T	T	A	C	G	G
-	0	0	0	0	0	0	0	0	0
G	0	0	3	1	0	0	0	3	3
G	0	0	3	1	0	0	0	3	6
T	0	3	1	6	4	2	0	1	4
T	0	3	1	4	9	7	5	3	2
G	0	1	6	4	7	6	4	8	6
A	0	0	4	3	5	10	8	6	5
C	0	0	2	1	3	8	13	11	9
T	0	3	1	5	4	6	11	10	8
A	0	1	0	3	2	7	9	8	7

best alignment score: 13

C T T - A C
G T T C A C
3 6 9 7 10 13

time complexity of DP approach sequence alignment.

* time complexity / time grow with the number of sequences
  * for two sequences $O(n^2)$
  * " three " $O(n^3)$
  * " k " $O(n^k)$

Alternative approaches

→ branch and bound
→ Heuristics
→ Dynamic programming table centred along the diagonal.

Heuristic approaches to MSA:

→ Heuristic methods
- ↳ Star alignment
- ↳ Progressive alignment methods
  - * CLUSTALW
  - * T-Coffe
  - * MUSCLE
- ↳ Heuristic variant of DP approaches
- ↳ Genetic algorithm
- ↳ Gibbs sampler
- ↳ Branch & Bound

→ Basic techniques
- * progressive
- * iterative
- * profile based
- * Divide & Conquer

Recurrence for 3 sequences

**

Align(S_i^1, S_j^2, S_k^3) → n sequences $(2^n - 1)$ recurrence

* $2^3 - 1 = 8 - 1 = 7$ equation

$$V_{i,j,k} = \max \begin{cases} V_{i-1,j-1,k-1} + S(a_i, a_j, a_k) \\ V_{i-1,j,k-1} + S(a_i, -, a_k) \\ V_{i-1,j-1,k} + S(a_i, a_j, -) \\ V_{i,j-1,k-1} + S(-, a_j, a_k) \\ V_{i-1,j,k} + S(a_i, -, -) \\ V_{i,j-1,k} + S(-, a_j, -) \\ V_{i,j,k-1} + S(-, -, a_k) \end{cases}$$

For K sequences dynamic programming table will have size n^k

Sum of pairs (SP) Score:

* Consider all pairs of letters in each column
* add the scores

SP score $\begin{pmatrix} A \\ V \\ V \\ - \end{pmatrix}$ = 2×Score(A,V) + Score(V,V) + Score(A,-) + 2×Score(V,-)

NB K sequences gives $\frac{K(K-1)}{2}$ addends

Score(-,-) = 0

Example

For the following alignment determine the SP score. consider match score = 2 and mismatch/gap penalty = −2

S1 = ACG--GAGA
S2 = -CG-TTGACA
S3 = ACH T-GA-A
S4 = CCG TTCAC-

Ans For column 1

SP score = 2S(A,-) + 2S(AC) + S(C,T) + S(A,A)

= (2+(-2)) + 2(-2) + (-2) + 2

= (-4) + (-4) - 2 + 2

= -8 //

For column 2

SP score (C,C,C) = 3S(C,C)

= 6 × 2 = 12

$\frac{4\times3}{2}$ $\frac{3\times2}{2}$ $\frac{2\times1}{2}$

SP Score (G, G, -, G) = 3 S(G, G) + 3 (G, -)

= 3 × (2) + 3 × (-2)

= 6 - 6 = 0

SP-Score (-, T, T, T) = 3 S(T, T) + 3 (-, T)

= 3 × (2) + 3 (-2)

= 6 - 6 = 0

SP Score (-, T, -, T) = 2 × S(T, T) + S(-, -) +

+ 4 (T, -)

= 2 + 0 + (-8)

= -6

SP-Score (G, G, G, C) = 3S(G, C) + 3S(G, C)

= 3 (2) + 3 (-2)

= 6 - 6

= 0

SP-Score (A, A, A, A) = 6S(A, A)

= 6 × 2

= 12

SP-Score (G, C, -, C) = 2S(G, C) + S(C, C)

+ 2S(G, -) + S(G, -)

= (-4) + 2 + (-9) + (-2) = -8

3+2/2

SP-Score(A, A, A, –) = 3(A, A) + 3(A, –)

= 6 + 3(–2)

= 6 – 6 = 0

Total SP-Score = –8 + 12 + 0 + 0 – 6 + 0 + 12 – 8 + 0

= 24 – 22

= 2

Evaluating alignment method:

Given an estimated alignment and a true alignment, compute various statistics (based on homology)

* SPFN: Sum of the false negative homology pairs
* SPFP: Sum of the false positive homology pairs
* TC: Total column score. Number of Columns. Columns that are identical in the two alignments
* Compression: Ratio of the estimated alignment length to true alignment length.

→ SP score = 1 - SPFN → measure of recall

→ Modeller score = 1 - SPFP → measure of precision

Error rate: SPFN, SPFP

Accuracy rate: TE score, SP-score, modeller score;

tree → alignment (iteratively)

Progressive alignment:

→ Start with a binary tree (merge tree/ guide tree) from the input set of sequences.

→ multiple sequence alignment is built from the bottom-up

→ Starting with internal nodes that have only leaves as children.

→ Interactively align the alignments to obtain an alignment for larger number of sequences.

Example

- Clustal
- clustal w
- clustal-Omega

→ Phylogeny

- Prank-aware
- MAFFT
- PAGAN

* Prank → PAGAN → Canopy

improve the accuracy of Prank, PAGAN

Problem : Create a bias tree that produce bias alignment.

* Doesn't realign the sequences
* Not guaranteed to ~~converge~~ converge to the optimal solution.
* the final alignment will be less accurate
  → if we have poor initial alignment.

Iterative alignment:

* Generate an initial MSA using a method like progressive alignment.
* Iteratively improves the MSA

Example:

SATé, SATé-2, PASTA

process terminate

* after fixed number of iteration
* maximum amount of time

Complexity

For N sequences, running time

$O(e^{2n})$

April 2020

Find the optimal local alignment of the following two sequences using -2 as the gap penalty, -1 as the mismatch penalty, and 2 as the score for a match. You have to use the Smith-Waterman algorithm. Show the DP table and mark the path, which corresponds to the alignment in the DP table.

TCAATG
ACATG

Sol^n:

Given

S_1 = TCAATG

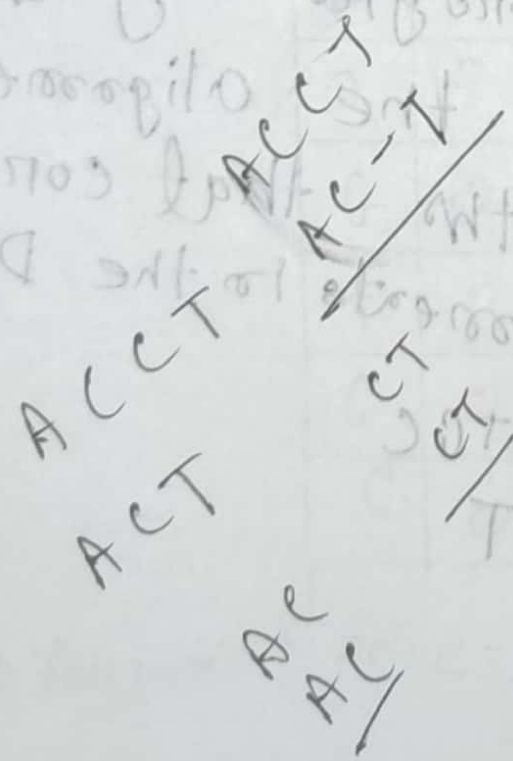
S_2 = ACATG

$S(x,y) = -1$; $S(x,x) = 2$

$S(x,-) = -2$

S1 ↓ / S2 →		A	C	A	T	G
	0	0	0	0	0	0
T	0	0	0	0	2	0
C	0	0	2	0	0	1
A	0	2	0	1	2	0
A	0	2	1	2	3	1
T	0	0	1	0	1	2
G	0	0	0	0	2	6

∴ highest alignment score = 6



A T G
A T G = 6
2 2 2

C - A T G
C A A T G = 6
2 -2 2 2 2

April 2021

Find the best (in terms of the alignment score) shortest and longest local alignments of the following two sequences using gap penalty of -2, a mismatch penalty of -1 and a match score of 2.

You have to use the Smith-Waterman Algorithm, and show the corresponding dynamic ~~pg~~ programming (DP) table. Please show the alignments and marks the paths that correspond to the alignments in the DP table

CCAGTGC

CTAT

Given that,

$S_1 = CCTAGTGC$

$S_2 = CTTATT$

$\delta(x,y) = -1$

$\delta(x,x) = 2$

$\delta(x,-) = -2$

$S_1 \downarrow$ $S_2 \rightarrow$		C	T	A	T
	0	0	0	0	0
C	0	2	0	0	0
C	0	2	1	0	0
A	0	0	1	3	1
G	0	0	0	1	2
T	0	0	2	0	3
G	0	0	0	1	1
C	0	2	0	0	0

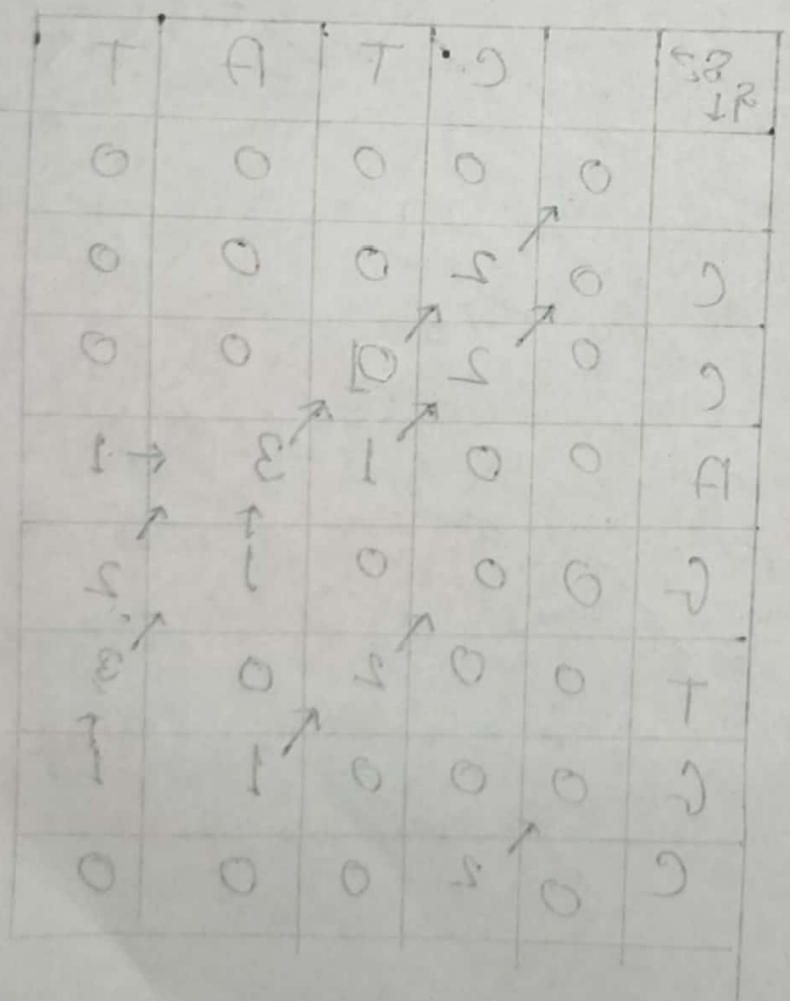
Ans: Alignment score = 3

Shortest best alignment:

C T A
C C A
2 -1 2 = 3

Longest best alignment

C T A _ T
C C A G T
2 -1 2 -2 2 = 3



Genome sequencing:

Determining the order of nucleotides or bases in a DNA molecule/gene

Genome assembly

Process of putting a large number of short DNA sequences (read) back together to recreate the original chromosome from which the DNA originated.

OR,

It refers to aligning and merging fragments from a longer DNA sequence in order to reconstruct whole genome

→ Human genome: 3 billion base pairs
→ small bacterial genome: few million base pairs.

Why challenging?

Ans: * Sequencing machines can't read whole genome at a time.

* It can determine small fragments called reads.

read: Sigle observation of the (partial) sequence of a DNA molecule

shotgun: Random fragmentation of the whole genome, like it was fired from a shotgun

Contig: Contiguous sequence of sequence often derived from multiple reads.

Scaffold: Linearly ordered and oriented group of contigs.

Steps in genome sequencing:

* Start with many copies of genome
* Fragment them and read one or both ends
* Find overlapping reads → overlap or Cde Bruijn graph
* merge them into contigs
* Scaffold contigs using paired end reads.

Coverage: It refers to the number of reads covering a particular position in the genome.

Average coverage: Average number of reads covering a position in the genome

types of coverage:

→ small coverage (shallow sequencing)

→ High coverage (Deep sequencing)

Basic challenges in genome assembly:

*two main challenges

→ getting sufficient coverage of the genome (read length, number of reads, size of genome etc.)

→ Assembling the reads into a complete genome.

Issues:

↳ coverage (more is good)

↳ Errors in reads (error by sequencing machine)

↳ genome are double standard.
Reads vary from very short (35b
to quite long (800bp)

↳ Repeats (repeat length > read length)

↳ Running time and memory

↳ Double stand

Sequencing error:

Mistakes made by sequencing machin-
es.

Repeat: Some portion of genome can be repeated more than one time in the whole genome.

Problematic when

* repeats are longer than read lengths

* repeats are present in many copies

Read length matters:

* Reads much longer than repeats → Assembly should be easy
* Reads equal to the repeats → Assembly computationally difficult (NP-hard)
* Read shorter than repeats → Assembly underdetermined

* Human genome is ~50% repetitive

Coverage

More coverage is good for better assembly

Summary of challenges:

Types of genome assembly:

→ De novo assembly → Do everything from scratch

→ Reference based / Comparative assembly → we have a reference genome

Comparative assembly? easier than De Novo ☆☆

→ take the reads
→ map them onto the reference genome
→ Collect all overlapping reads
  ↳ produce a MSA
  ↳ produce consensus sequence

Advantages:
* Fast → highly accurate even when have errors
* Short reads can map to several places (if they have errors)

Disadvantages:
* Needs close reference genome
* repeats are problematic.

De novo assembly:

→ Much easier to do with long reads

→ Need very good coverage

→ Necessary when you don't have a closely related (correctly assembled) reference genome

De Novo Assembly paradigms:

* Overlap graph
* K-mer graph / De Bruijn graph

→ Overlap layout consensus methods
  → greedy (TIGR Assembler, phrap, CAP3)
  → graph based (Celera assembler, Arachne)

Shortest common superstring: (SCS)

input: given collection of strings

output: find SCS

* without requirement of shortest:
  * just concatenate input strings

String composition problem: (String → k mers)

Given a string text, its k-mer composition $\text{COMPOSITION}_k(\text{Text})$ is a collection of all k-mer substrings of Text. (included repeated k-mer)

Example:

$\text{COMPOSITION}_3(\text{TATGGGTGC}) = \{\text{TAT, ATG, TGG, GGG, GGT, GTG, TGC}\}$

There will be $(\text{len}(\text{Text}) - k + 1)$ k-mer.

NB: genome assembly is inverse problem

→ # K-mer from a text

String reconstruction problem

(k-mers → string)

Reconstructing a string from its k-mer composition.

input: an integer k and a collection patterns of k-mers

output: A string text with k-mer composition

* Connect a pair of k-mers if they overlap in k-1 symbols.

Example:

S = {AAT, ATG, GTT, TAA, TGT}

Simple concatenation

Super string length = 3×5 = 15

```
TAA
 AAT
  ATG
   TGT
    GTT
TAATGTT
```

∴ Super string length = 7

Example 2:

S = {AAT, ATG, ATG, ATG, CAT, CCA, GAT, GCC, GGA, GGG, GTT, TAA, TGC, TGG, TGT}

TAA
AAT
ATG

We can extend it using TGC TGG, TGT

TAA
AAT
ATG
TGT
GTT

if we extend by using TGT there is no other extension other than GTT.

extend by using TGC

TAA
AAT
ATG
TGC
GCC
CCA
CAT
ATG
TGG
GGA
GAT
ATG
TGT

Overlap graph:

It is a directed graph in which a vertex is a read, a directed edge is an overlap between suffix of source and prefix of sink. Weight of edges are number of overlap.

∴ in summary,

vertex → read

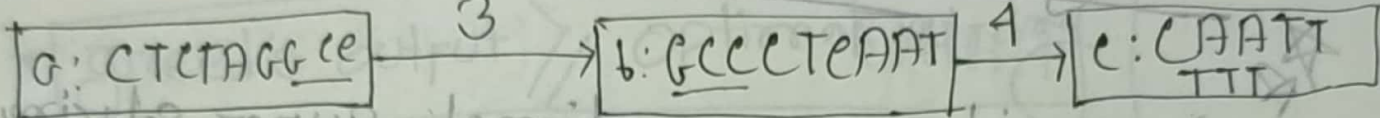
edge → overlap between suffix of source to prefix of sink

weight → number of overlap.

Example:

V = {a: CTCTAGGCG, b: GCCCTCAAT, c: CAATTTTTT}

edge = {(a,b); (b,c)}



How can we solve it (finding SCS from overlap graph)?

* modify overlap graph where,

$$\text{cost} = -(\text{length of overlap})$$

* SCS corresponds to a path that visits every node once, minimizing total cost along path.
* So it is nothing but a Traveling Salesman Problem (TSP)

∴ It is NP-hard

If we disregard edge weights and just look for a path that visits all the nodes exactly once,

→ Hamilton path problem (NP-complete)

Greedy Shortest Common Superstring:

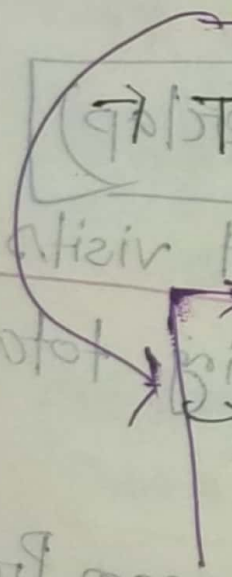
→ provide non-optimal superstring

→ greedy algorithm chooses longest overlap between two reads.

→ Ties are resolved arbitarily

→ Concatenate two strings

→ Update graph



Failure scenario of Greedy SCS

* not guaranteed to choose, overlap yielding SCS
* It's have a good approximation ratio (ratio ~ 2.5)

i.e Superstring by greedy SCS won't be more than 2.5 times longer than the true SCS

* It may results more than one fragment

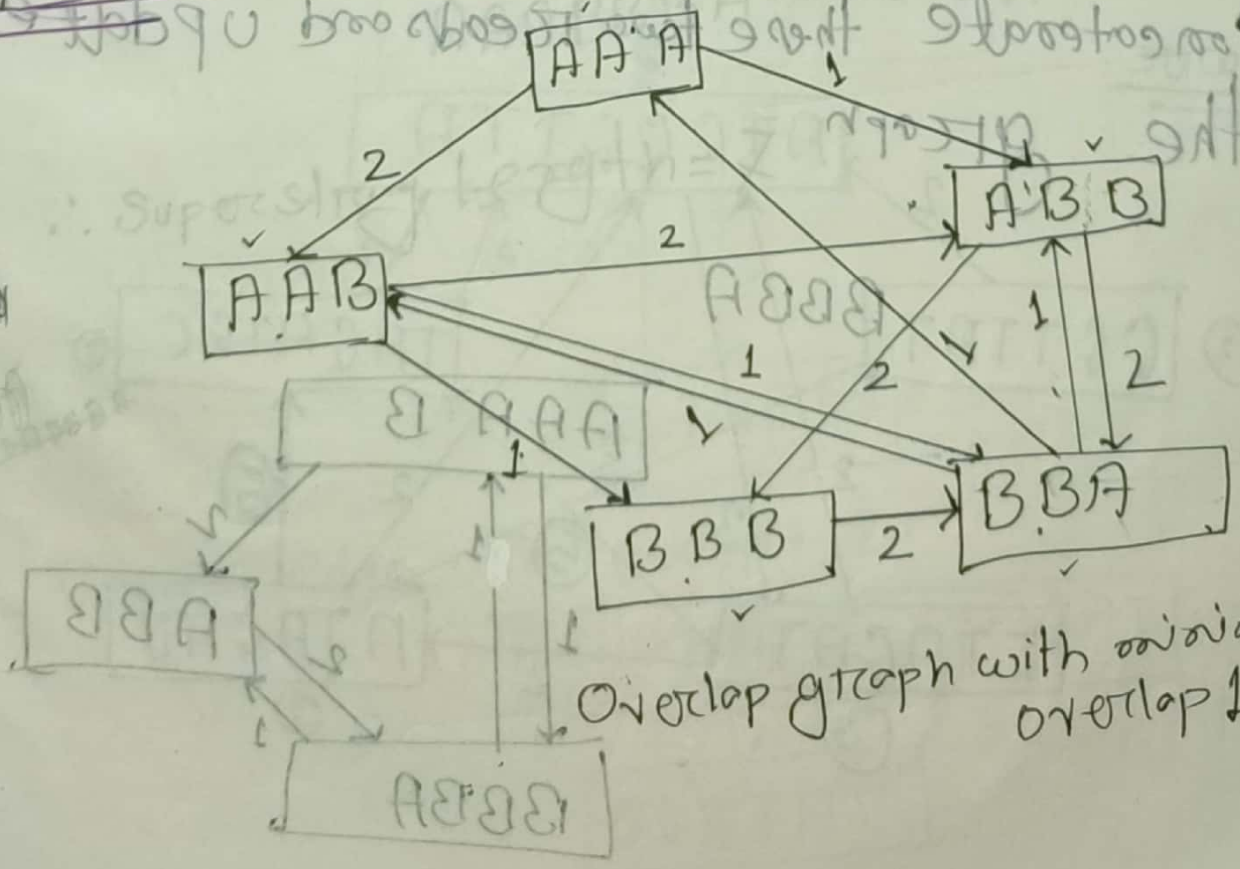
* greed's output ≥ optimal string

Example: Construct overlap graph with minimum overlap 1 and find a shortest common superstring from that gra.

Ph: AAA AAB ABB BBB BBA

S: AAABBBA

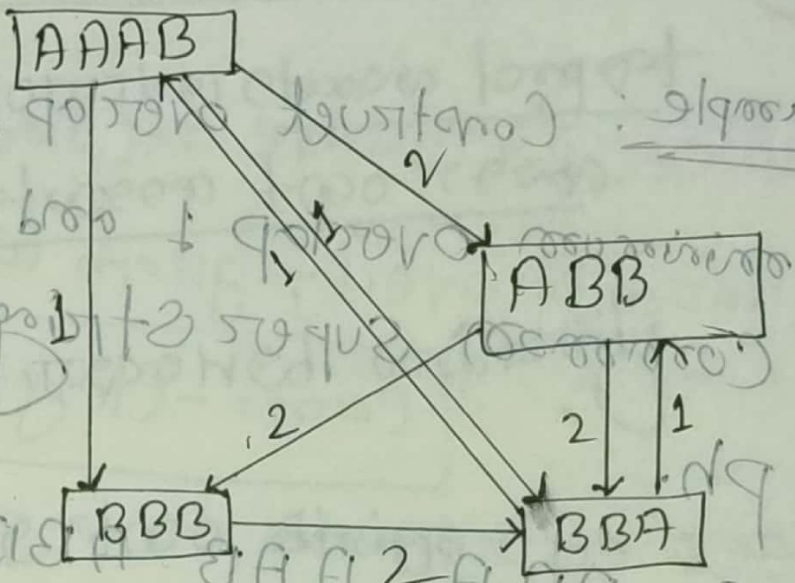
Answer:



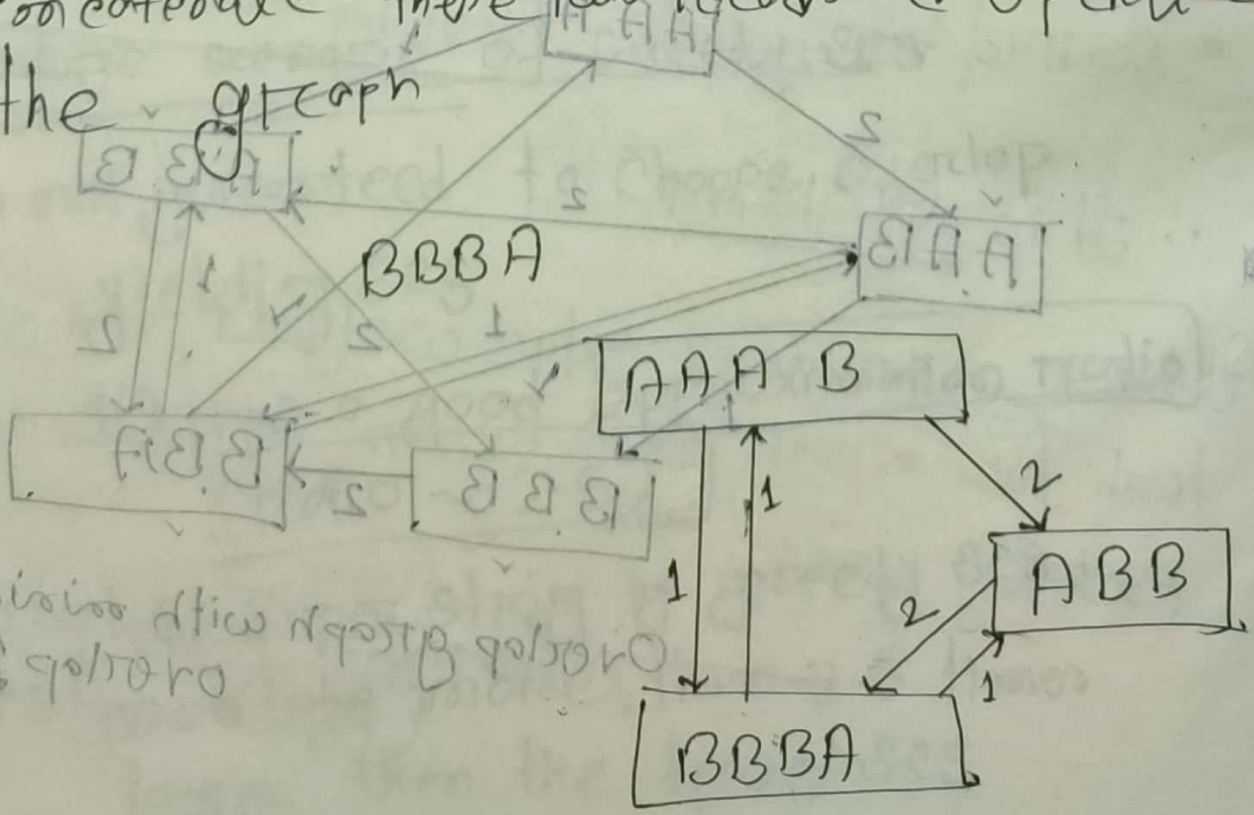
Overlap graph with minimum overlap 1.

Choose edge (AAA → AAB)
Concatenate these two strings and update the g

AAAB

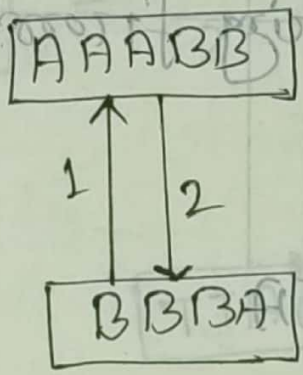


Choose edge (BBB → BBA)
Concatenate these two reads and update the graph



Choose edge (AAAB $\xrightarrow{2}$ ABB) and concatenate these two strings.

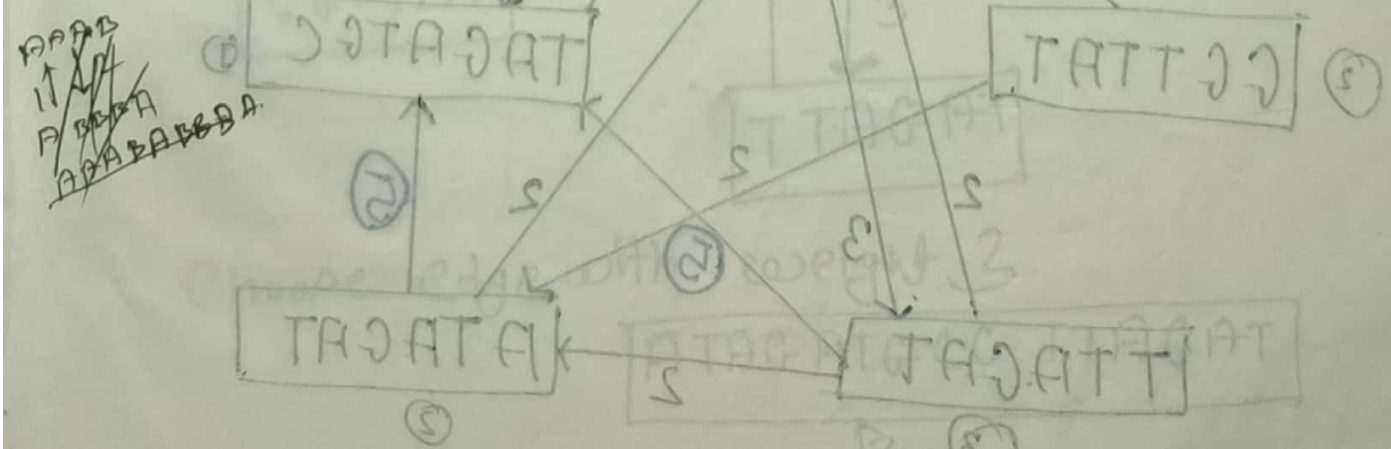
AAAABB



Choose edge (AAABB $\xrightarrow{2}$ BBBA) and find final string.

AAABBBA

∴ Superstring length = 7



April 2021

Example Consider the following five reads.

Construct the overlap graph with minimum overlap 2. Find a shortest common superstring from that overlap graph.

ATGCGTTA

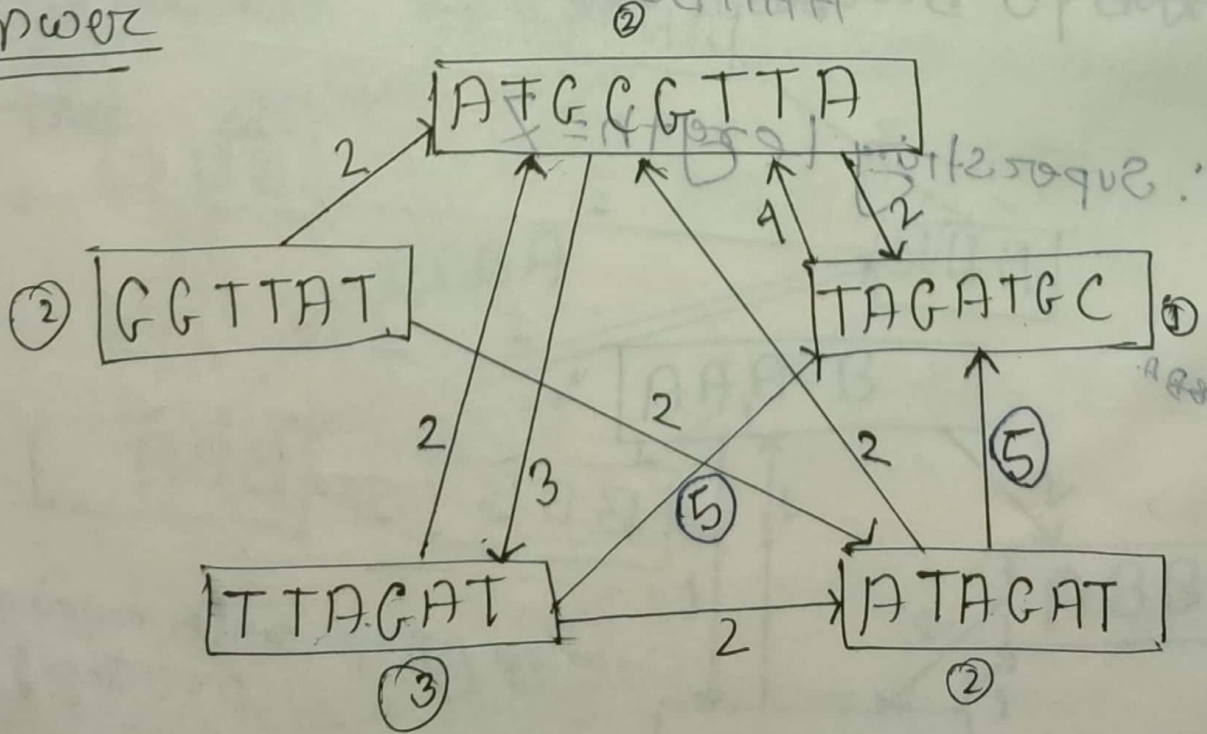
GGTTAT

TAGATGC

TTAGAT

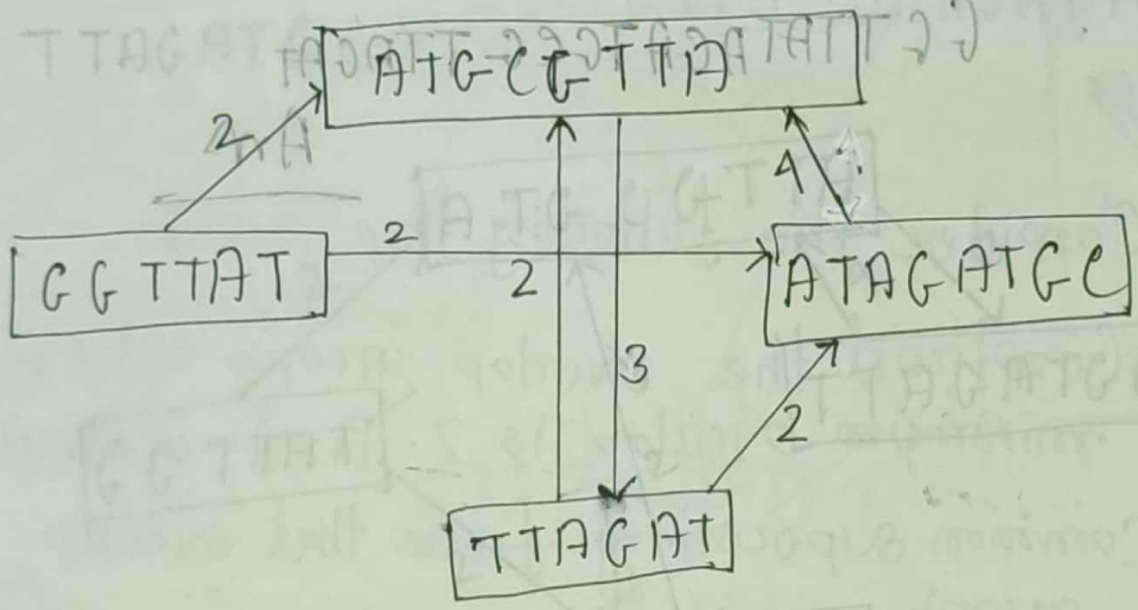
ATAGAT

Answer

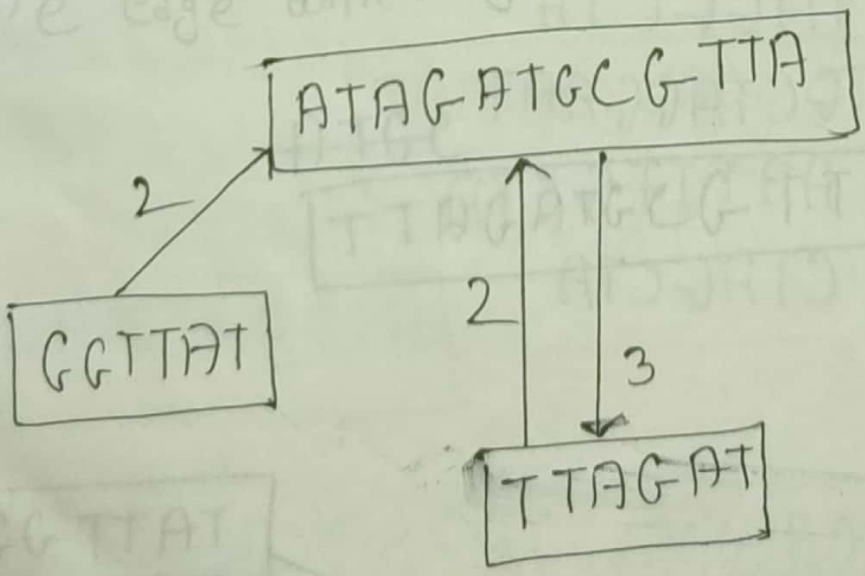


Approach -1

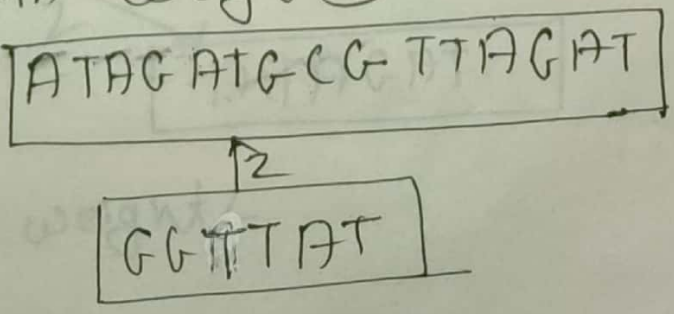
Choose edge ATAGAT $\xrightarrow{5}$ TAGATGC



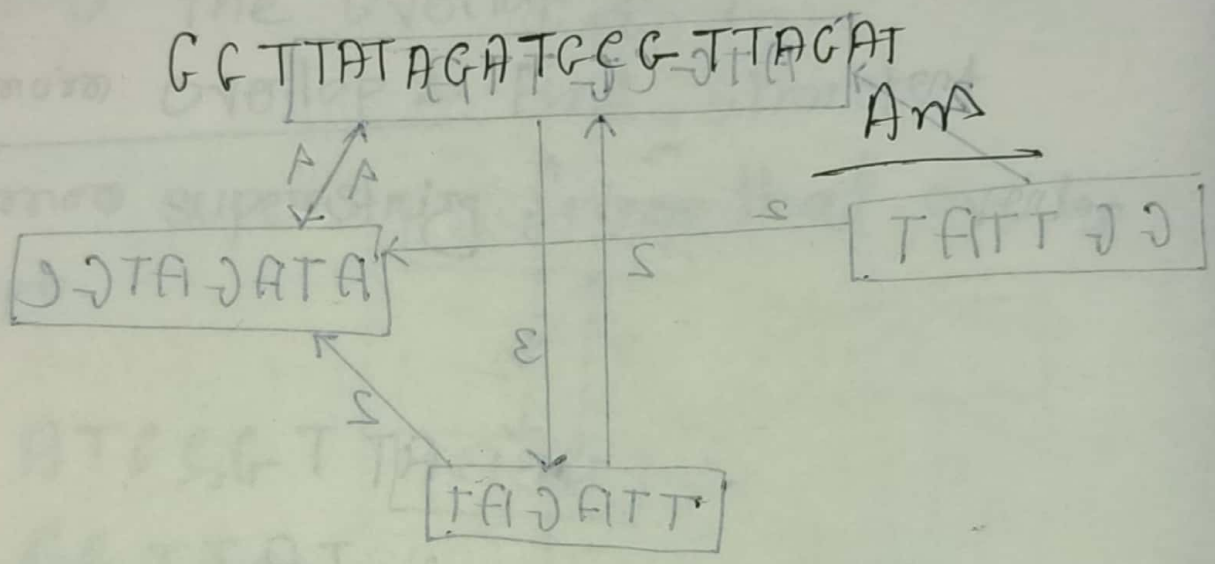
Choose edge with weight 1

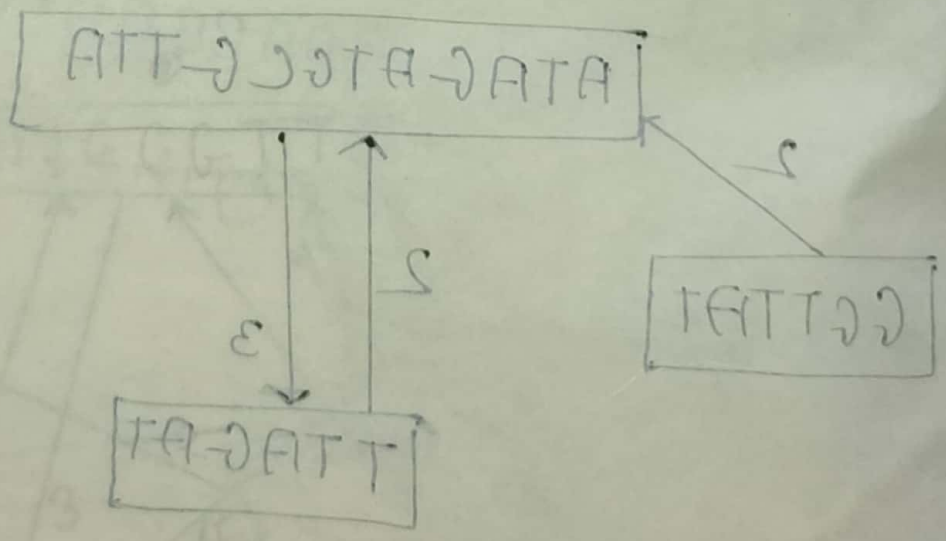


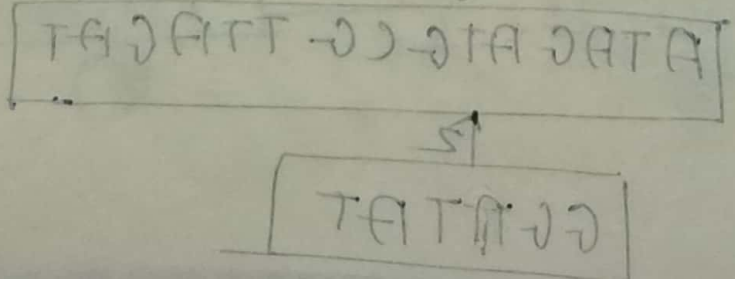
Choose edge with weight 3



Choose edge with weight 2



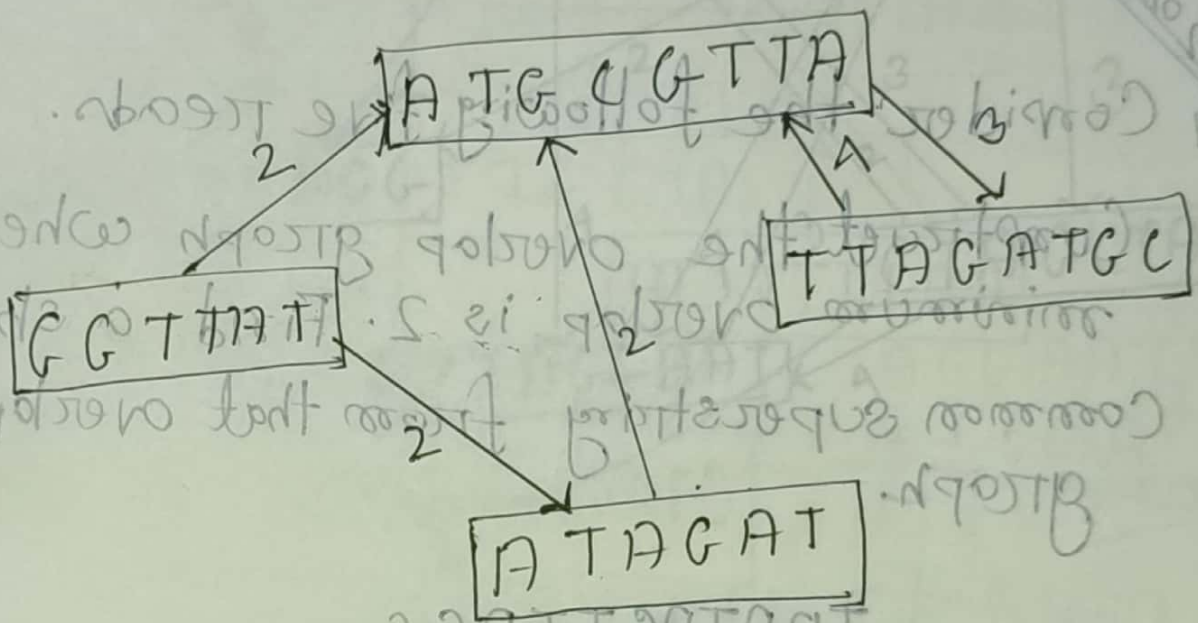




Approach 2

Choose edge (TTAGAT $\xrightarrow{5}$ TAGATGC)

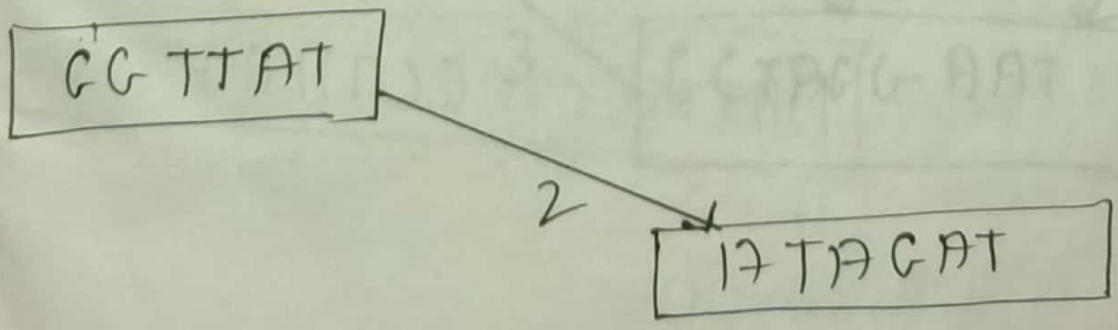
TTAGATGC



Choose edge with weight 1. and concatenate.

~~ATGC~~

TTAGATGCGTTA



Choose edge with weight 2

GGTT ATAGAT

TTAGATGCCTTA

April 2020

Consider the following five reads.

Construct the overlap graph when minimum overlap is 2. Find a shorter common superstring from that overlap graph.

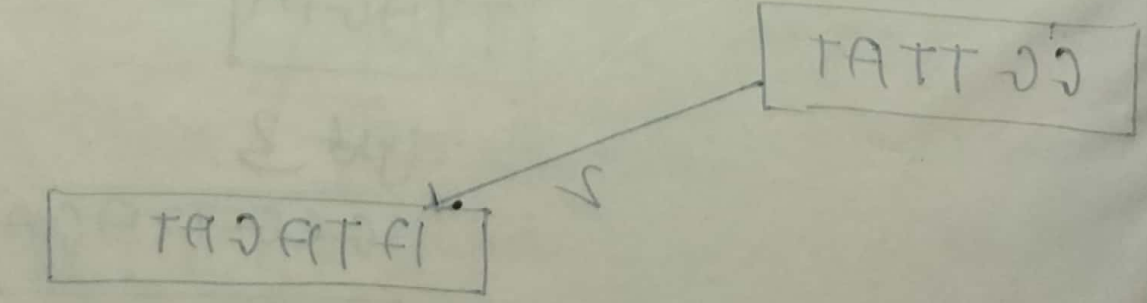
TAATACTTAGG

TAGGCTA

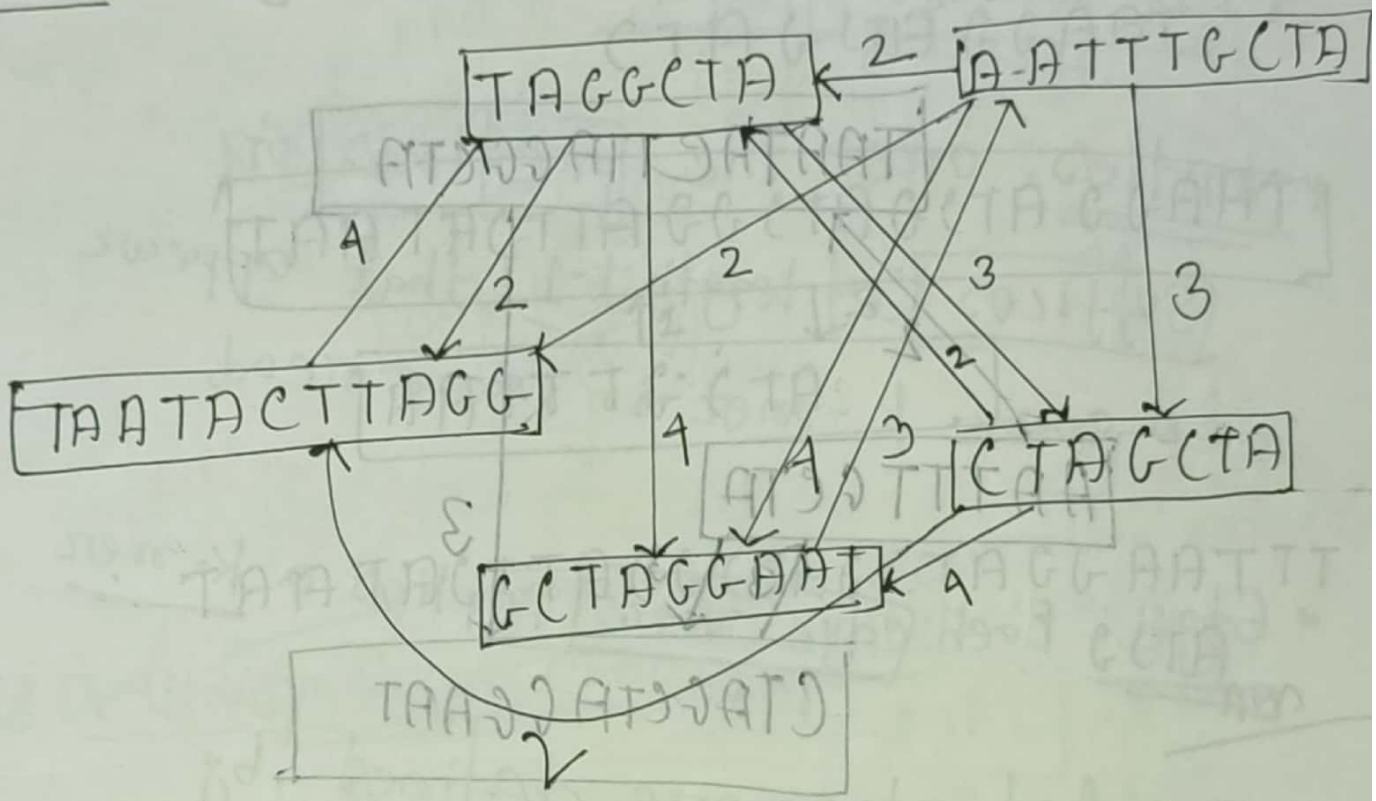
GCTAGGAAT

AATTTGCTA

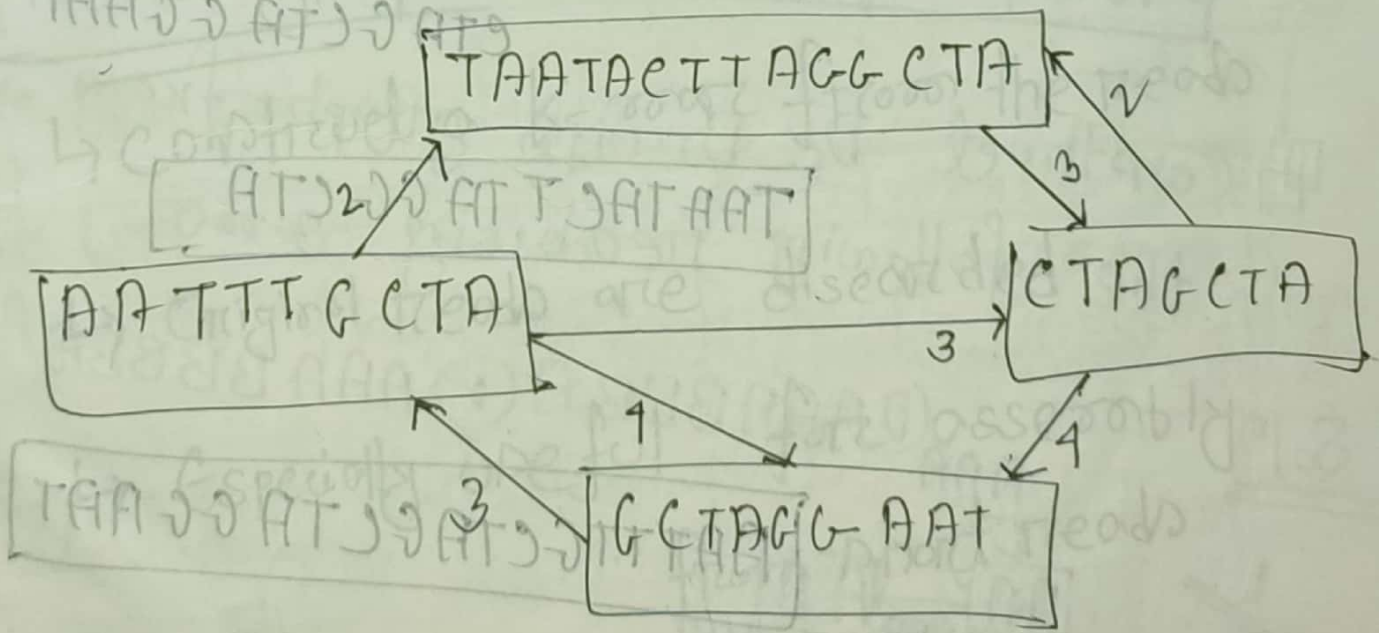
CTAGCTA



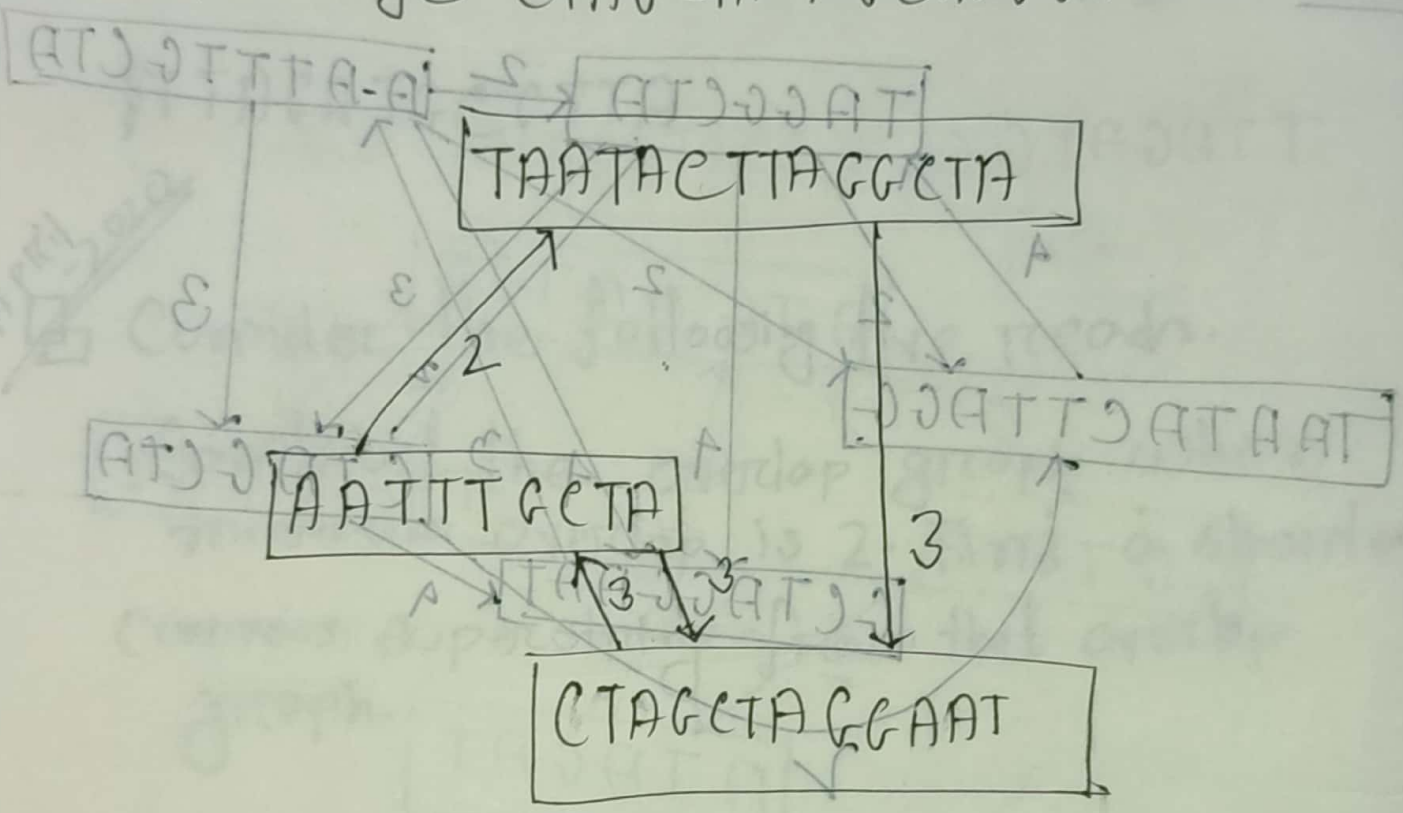
Answer



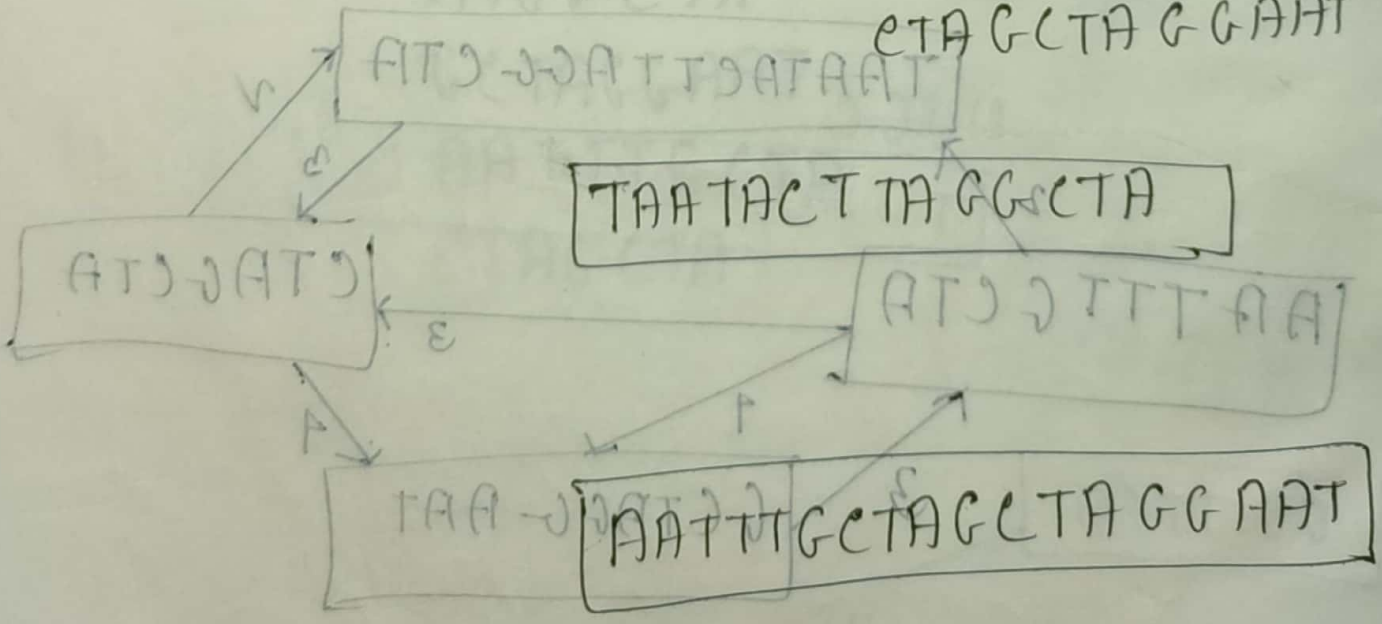
Choose ~~A~~ TAATACTTAGG $\xrightarrow{4}$ TAGGCTA



Choose edge CTAGCTA $\xrightarrow{A}$ GCTAGGAAT

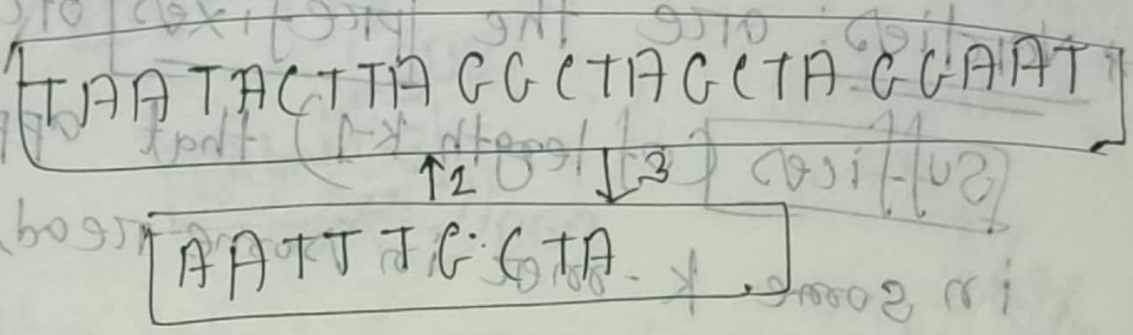


Choose edge with AATTTGCTA $\xrightarrow{3}$ CTAGCTAGGAAT



Choose edge with TAATACTTAGGCTAGCTAGGAATT → 3

CTAGCTAGGAAT



TAATACTTAGGCTAGCTAGGAATTT

CTA

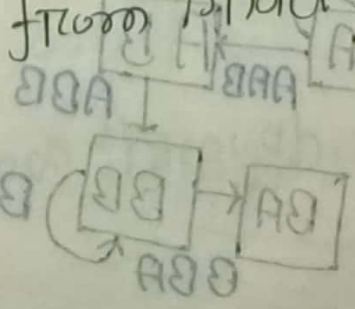
Am

K-mer graph

↳ Construct a K-mer from the reads

↳ Original reads are discarded

↳ Especially useful for assembly from short reads



De Bruijn graph

* Vertices: are the prefixes or suffices (of length, k-1) that appear in some k-mer in some read,

* Edge: Each edge v→w implies a k-mer

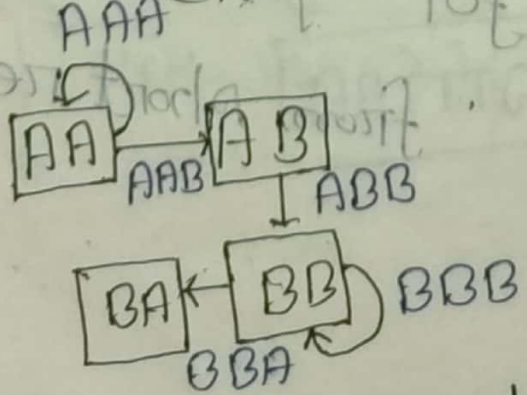
- Directed edges are defined by overlap of k-2 nucleotides

Construct De Bruijn graphs from the following reads using 3-mer

(1) AAABBBA (2) AAABBBBA

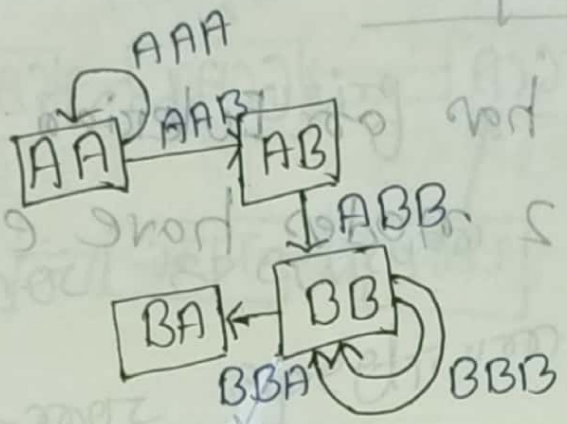
Soln

1



* Each edge corresponds to a length-3 input string.

11 Given read: AAABBBBA, K=3



De Bruijn Graph Characteristics:

- Small value of K produce small graph → sensitive to sequencing errors
- Doesn't require all pairs overlap calculation
- Also produces fragmented assemblies
- Long range connectivity information is lost for small K

Assembly / finding string from De Bruijn graph

→ by using Eulerian path.

Eulerian path:

Path that goes through every edge exactly once.

Condition of Eulerian path

undirected graph

* if a graph has an Eulerian path the all but 2 nodes have even degree.

directed graph

* A graph has an Eulerian path if and only if the

indegree(v) = outdegree(v) for all but 2 nodes (x, y) where

indegree(x) = outdegree(x) + 1
indegree(y) = outdegree(y) - 1

failure scenario of De Bruijn

* Need to know the correct order of k-mers if there is are repeats in the string

April 2024

Construct De Bruijn graph from the following read using

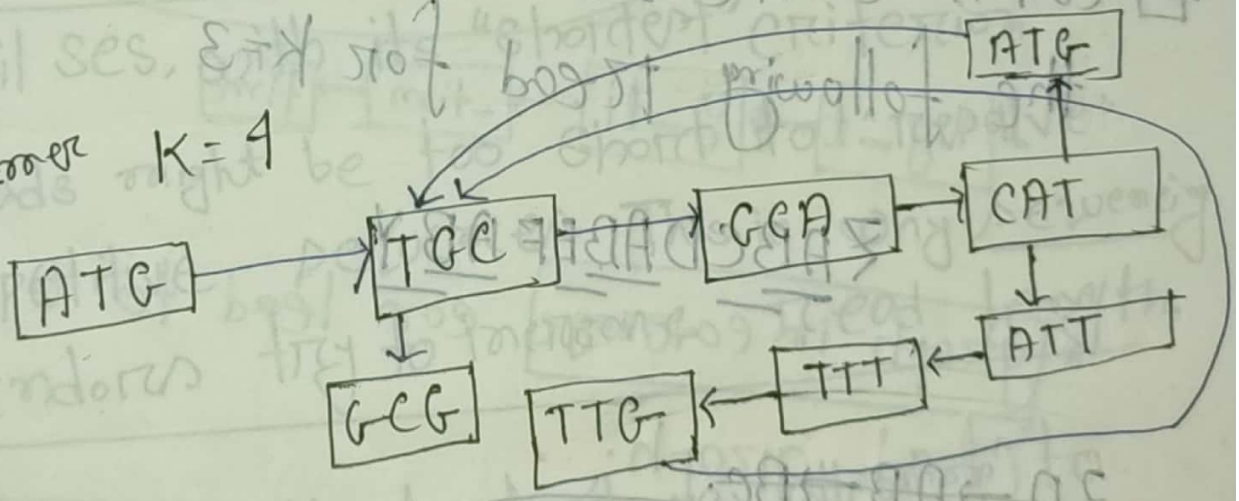
i) 3 mer spectrum

ii) 5 mer spectrum

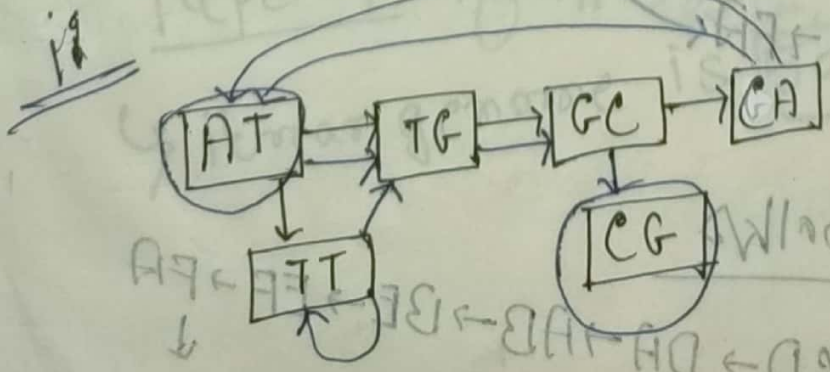
ATGCATTTGCATGCG

Soln

i) 4-mer K=4

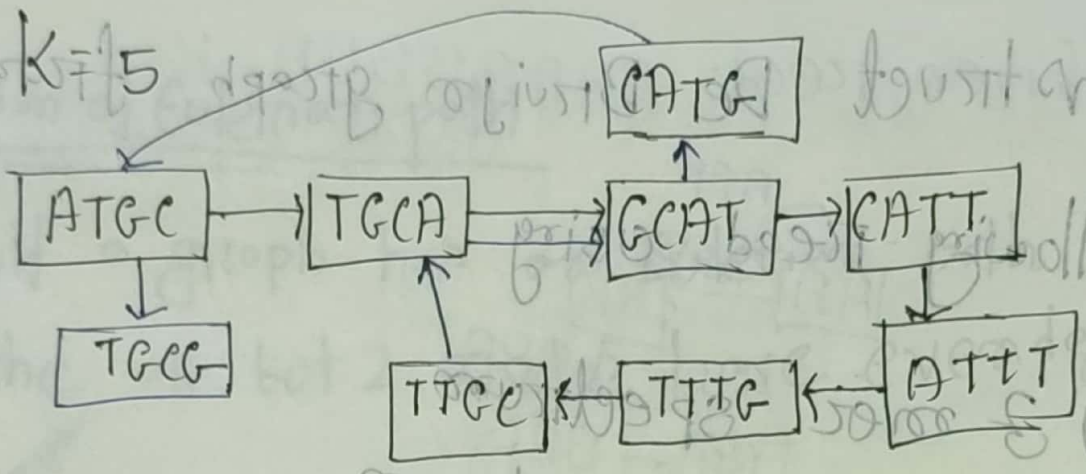


ii)



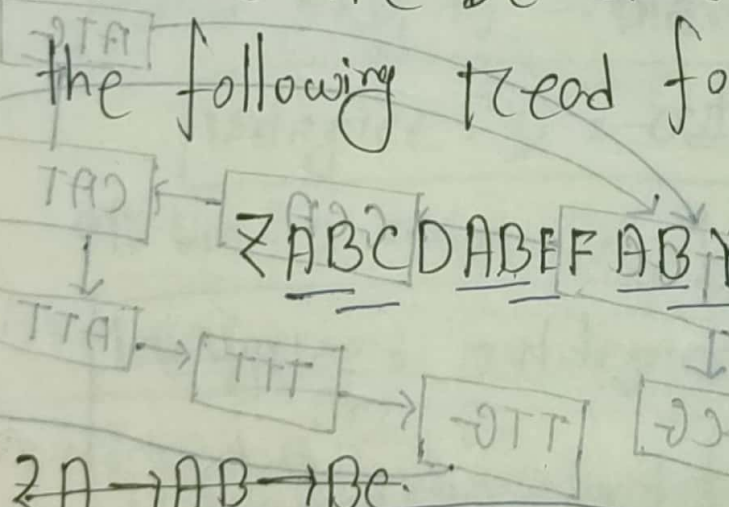
K=3

iii K=5

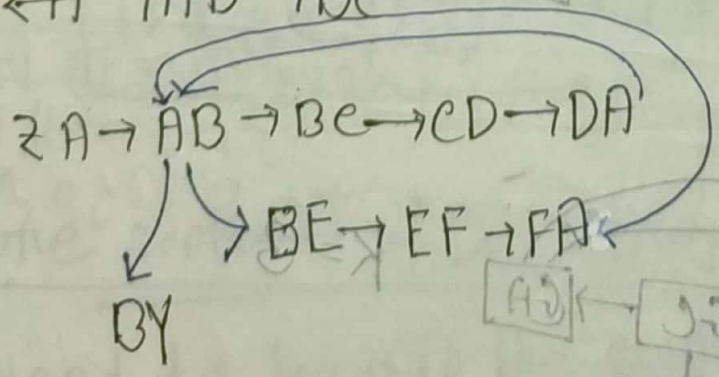


Construct the De Bruijn graph from the following read for K=3

ΣABCDABEFABY



ΣA→AB→BC



Eulerian paths/walks

ΣA→AB→BC→CD→DA→AB→BE→EF→FA
↓
BY ←AB

ΣA→AB→BE→EF→FA→AB→BC→CD→DA
↓
BY ←AB

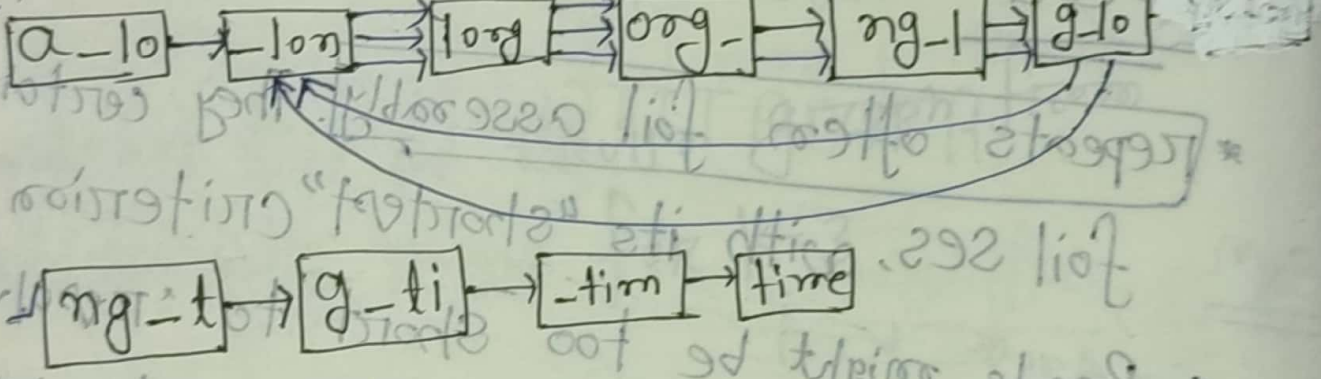
These correspond to two disjoint directed edge cycle joined by node AB.

Repeat

* repeats often fail assembly. They certainly fail SES, with its "shortest" criterion
* Reads might be too short to resolve repetitive sequences. This is why sequencing vendors try to increase read length.
* Algorithms that don't pay attention to repeats might collapse them.
  ↳ Human genome is 50% repetitive

Compute the De Bruijn graph for the following string with k = 5.

a_long_long_long_time



* graph is coverage can lead to a disconnected graph.
* connected components are individually Eulerian, overall graph is not.

Genome Rearrangement

Process by which the order of genes, DNA segments, or even entire chromosomes in an organisms genome are altered or rearranged

↳ Turnip and cabbage share a common ancestry

↳ Human and mouse " a common..

- they share the same genes but in different order
- A series of rearrangement
  → transforms one genome into others the

~~Mouse and~~

Genome Rearrangement:

Mouse and human X chromosomes represented as eleven colored directed segment (Synteny block)

↳ Most human genes have a mouse counterpart

↳ But in different order

↳ Hundreds of genes often appear one after another (as a group: synteny) but may have different order in different species

Transforming Mouse genome to Human genome (Sorting by Reversal)

Sorting by Reversals

$\pi_1 = 5\ 1\ 4\ 3\ 2\ 6\ 7\ 8\ 9\ 10$

$\rho(1,2)$

$\pi_2 = 1\ 5\ 4\ 3\ 2\ 6\ 7\ 8\ 9\ 10$

$\rho(2,5)$

$\pi_3 = 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9\ 10$

Common Rearrangement Practical implication:

* Mouse provides insight into Human Genetic Disorder.

Waardenburg's Syndrome

↳ Gene implicated in the disease was linked to human chromosome 2

↳ A breed of mice had similar symptoms caused by the same type of gene as in human

↳ Scientist succeeded in identifying location of gene responsible for disorder in mice

↳ Finding the gene in mice gives clues to where the same gene is located in humans

GWAS (Genome Wide Association Studies)

Genetic Association:
genotype → phenotype

* Attempts to detect how genotype affects phenotype in population.

Key components of association studies:

↳ measure genetic variation

↳ measure phenotype variation

↳ Quantify the association between genetic and phenotypic variations

↳ multiple organisms - cells, etc.

Heritability:

Extent to which a trait is predictably passed from generation to generation

* Phenotypic variation = genetic + environment

genetic + environment

→ Down's syndrome	→ Heart disease
→ Huntingtons Disease	→ Height
→ 3rd copy of Chromosome 21	

→ Consider potential confounders

Case and Control design

take 200 peope genome sequence

Case: 100 people have disease/trait

Control: 100 " " the " no "

* other traits are almost same (weight age, height)
* notified the different portion between two group of people genome

Confounding factor: If other traits are not similar, then an ambiguity will create

Ex:

Case: people of Bangladesh have Diabetic

Control: " of America have no "

* traits are genetic / environmental Some
* GWAS is used for genetic variant

Manhattan Plot → Association testing
↳ Association of each SNP with phenotype
↳ Statistical test (P-value)

→ Used in any GWAS Study
→ take snp of each chromosome
→ find out the likelihood against each snp
→ snip that cross a threshold are responsible (association Studies)

SNP: (Single Nucleotide Polymorphism)

Portion of genome that varies between two (people) agents (A_1, A_2)

→ find the corresponding gene against snp

A DNA sequence variation that occurs when a single nucleotide (A, T, C, G) in genome sequence is altered and the particular alteration is present in 1% of the population.

BLAST (Basic Local Alignment Search Tool)

↳ find regions of similarity between biological sequences.

↳ most widely used bioinformatics programs
- for user searching

↳ Replace Smith-Waterman full alignment algorithm with faster heuristics for local alignment

Alignment tool

→ SATe
→ PASTA
→ MAFFT
→ MUSCLE
→ Bootai

Gene tree estimation

- RAxML
- FastTree
- others: variant of RAxML

* Species tree from gene trees:

- wQFM
- ASTRAL (2019, highly used)
- MP-EST (Highly used before 15 years ago)
- BEAST (Can out perform ASTRAL, Not Scale)
- BUCKY
- Combined Analysis

DNA storage / DNA digital data storage:

Refers to the process of storing digital data in the base sequence of DNA

- Next generation storage

Main idea:

↳ convert file → into binary data

↳ binary data → DNA sequence

↳ DNA synthesis (DNA sequence → biological material)

↳ store (DNA)

↳ read DNA set by using sequencer

↳ Mapping to binary data

↳ find original data

DNA synthesis

Confounding variable

A variable that influences both the dependent variable and independent variable

→ causing a spurious association.

* Confounding is a causal concept and such can't be described in terms of correlations or associations.

→ They confound the true relationship between two variables.

Phylogenomics practical challenges

Improved species tree estimation in the presence of

- → Gene tree discordance
- → Missing data
- → Poor signal per gene
- → Large dataset

* Combined analysis → Concatenation provides the advantage of high level of signal
  → statically inconsistent
* Summary method → generate inaccurate species tree if gene tree is ''
  ↳ small length gene → inaccurate gene tree

discordance create model misspecification

Poor signal per gene

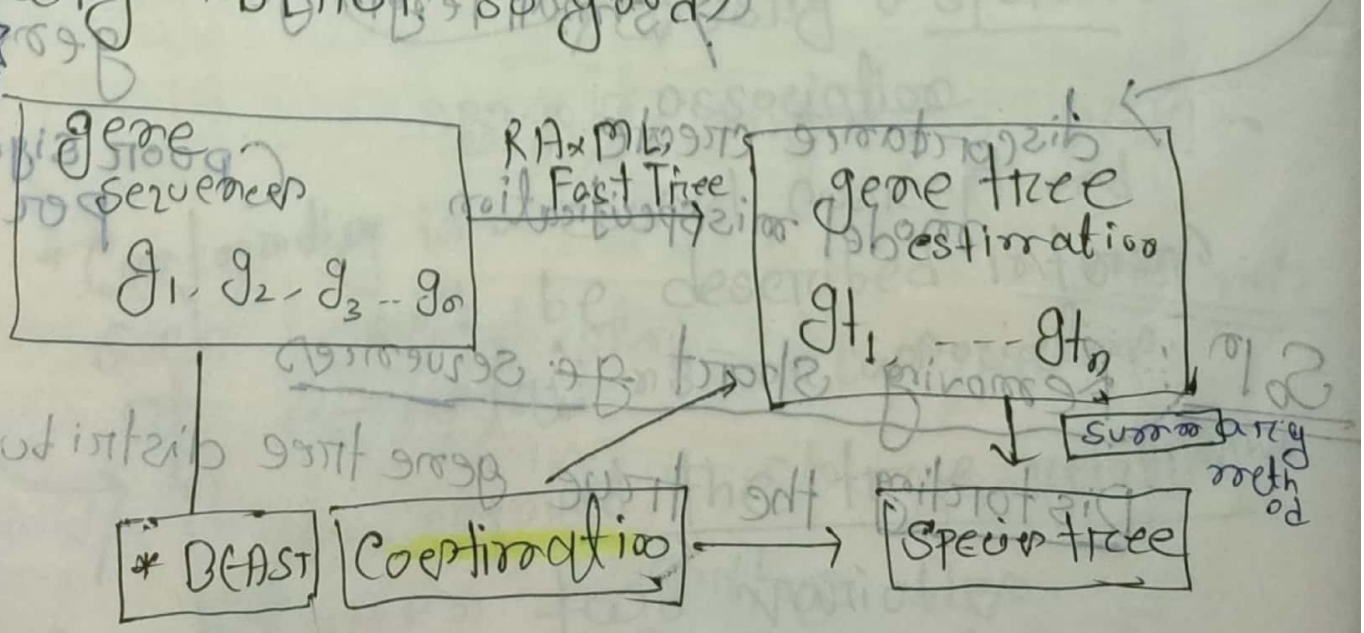
Sol^n: Removing short sequences

Distorting the true gene tree distribution

Species tree estimation methods

* Statically inconsistant
  - CA
  - Phylonet
  - MRP
  - GC
* Statically consistant
  - *BEAST (Coestimation)
  - MP-EST
  - BUCKy-pop

Why *BEAST so good?



Gene tree accuracy

* *BEAST produces much more accurate gene trees, than the most popular likelihood methods: (RAxML, FastTree)

> accuracy *BEAST > accuracy RAxML, FastTree

* Even the simplest methods like greedy consensus produce as good species tree as *BEAST given the gene trees estimated by *BEAST

NB: / finding:

Highly accurate species tree of *BEAST is largely due to the highly accurate gene tree.

↳

Problem:

Combined analysis → statically inconsistent

* BEAST → Not scalable

Summary method → : poor signal

Sol^n:

Combining the strengths of

Combined analysis

Summary method

3 approaches

↳ Naive binning

↳ Statical "

↳ Weighted Statical binning

Avian Genomes

Avian tree

Naive binning

→ Let we have n numbers of gene

→ take n numbers of gene into a bin randomly

→ Combine them

→ generate each gene of a bin super gene trees

→ generate species tree

* Naive binning improves the summary methods substantially

Problem

→ Model violation (Combining gene randomly)

Sol^n → Statistical binning

* Combining genes that are "similar"

→ Make a graph

* vertex → each gene tree by using incompatibility graph

* edge → if two vertex has certain level of discordance

Color graph

two types of bin
- balanced bin size
- unbalanced bins

* use graph color
* #bin = # colors

* unbalanced bins are problematic
* large bin size problematic due to undesirable properties of concatenation
* small bin would not provide increased signal

Theorem:

Phylogenomics pipeline using statistical binning and coalescent-based summary method is not statistically consistent

Statistical binning + coalescent-based summary method

weighted " " + statistically consistent

not statistically consistent

Coalescent-based summary method:

- MP-EST
- ASTRAL
- STAR
- NJst
- BEAST (Coestimation)

} probability based

Sequencing technology

technology	Read length	Error rate	Reads/run	time/run

April-2021

Suppose you are trying to construct a species tree on 15 different species. You have sampled 300 gene sequences from each of these 15 species.

Your supervisor has asked you to you to use a method called GTest for constructing trees from sequence alignment and sp-est (which is a summary method) for estimating species trees from gene trees.

i) How many times do you need to run GTest and SP-est to estimate a species tree by summarizing gene trees?

ii) How many times do you need to run GTest and SP-est to estimate a species tree by if you use "naive Binning" with 10 bins?

iii) How many times do you need to run GT-est and sp-est to estimate a species tree if you use "statistical Binning (SB)" assuming that SB returns 12 bins!

iv) How many times do you need to run GT-est and sp-est to estimate a species tree if you use "weighted statistical Binning (wSB)" assuming that wSB returns 12 bins!

v) How many times do you need to run GT-est and sp-est into estimate a species tree using Combined analysis!

Ans: Given

300 gene
15 species

i Species tree by summarizing gene tree

GT-ept: 300

SP-ept: 1

ii Species tree by using "Naive Binning" with 10 bins.

GT-ept: 10

SP-ept: 1

iii Species tree by using "Statistical Binning (SB)" assuming that SB returns 12 bins

GT-ept: 12

SP-ept: 1

iv Species tree by using "Weighted Statistical Binning" assuming that WSB returns 12 bins:

GT-est: 300

SP-est: 1

v Species tree by using Combined analysis

GT-est: 1

SP-est: 0