

Computational Biology, Phylogenetic Trees

Consensus methods

Asger Bruun & Bo Simonsen

The 16th of January 2008

Department of Computer Science

The university of Copenhagen

0 Motivation

- Given a collection of Trees $T = \{ T_0, \dots, T_n \}$
- We want to find a common tree that combines all trees in T:
 - Different algorithms for constructing a phylogenetic tree gives different results, we want to combine those into one single tree.
- We may have different biological data from the each species (taxa), we will represent them, in a phylogenetic tree, using consensus methods

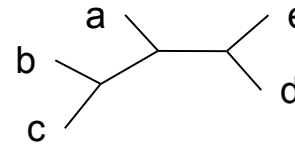
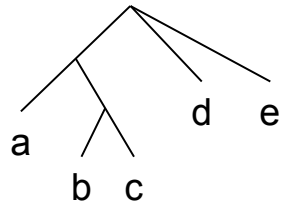
Methods

We'll consider these methods:

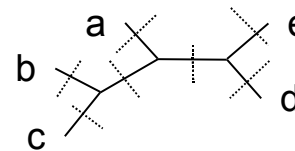
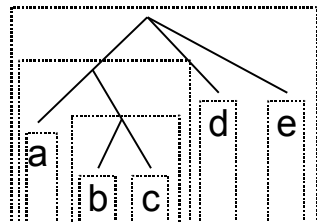
- All trees (input and output) have the same set of taxa
- All the input trees have a subset of taxa, but the output tree contains the set of taxa.

¹ Methods based on splits and clusters

In a phylogenetic tree (rooted or unrooted) every leaf represents a taxon

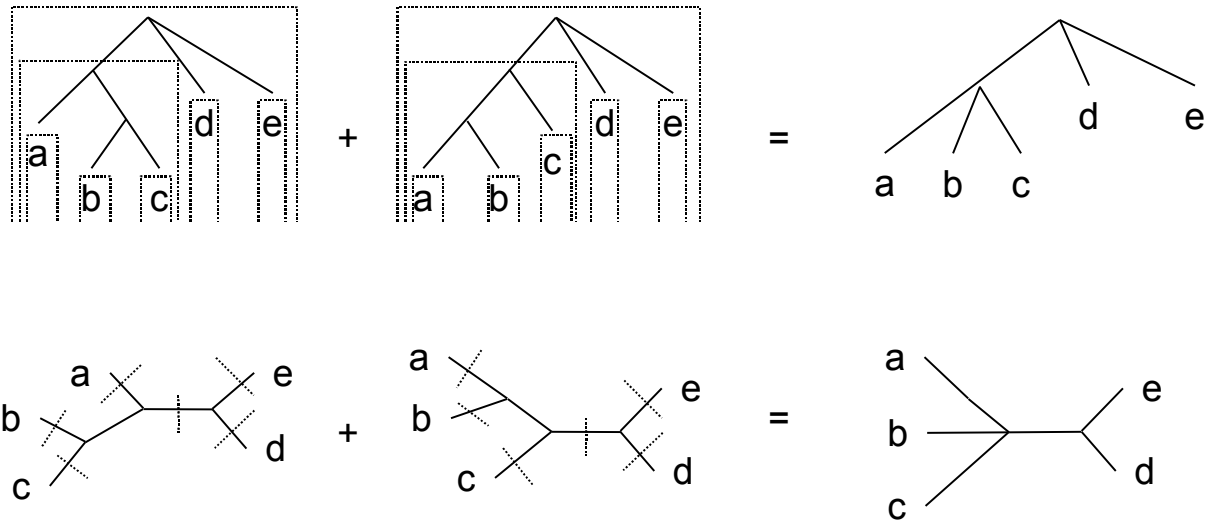


A phylogenetic tree can be subdivided into clusters (/monophyletic groups/clades) or splits



1.1 Strict consensus

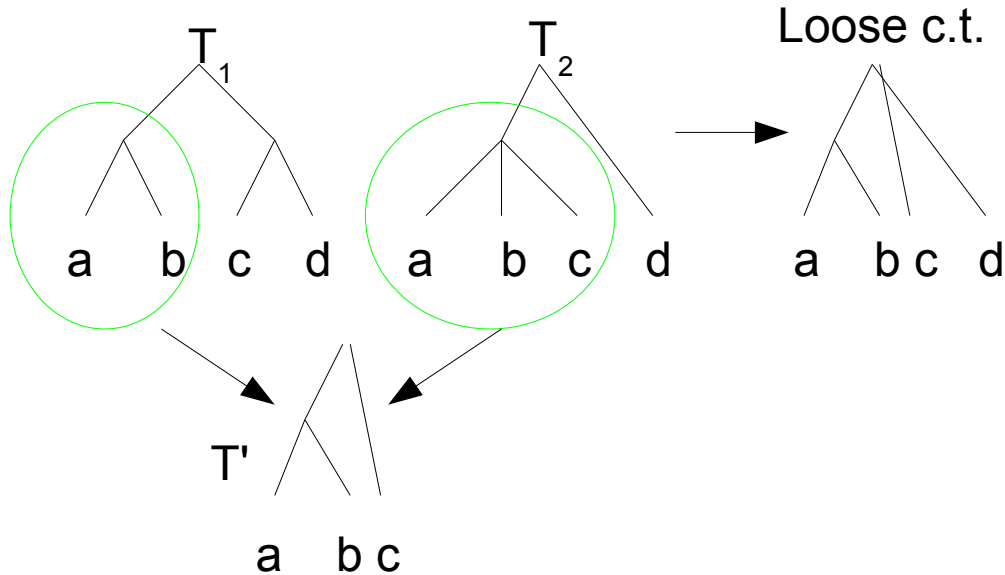
a simple method for tree consensus is: select the clusters or splits common to every input



1.2 More methods based on splits and clusters

Majority rule – like strict, but only select clusters / splits which is present in 50% of the trees in T

Loose consensus (aka semi strict) – select clusters / splits which are compatible with every tree in T .



Definition

A collection of groups C is compatible if there exists a tree T' s.t. every group in C is a cluster / split of T' .

Greedy consensus – Like the loose consensus tree, but input splits is sorted by frequency. Take the element with the highest frequency, and build a collection of compatible clusters / splits. This gives the greedy consensus tree.

2 Methods based on intersection

Adams consensus

- First of all consensus methods.
- Will only work on rooted trees, no analogue for unrooted trees
- We construct the tree recursively, using this algorithm:

```
Procedure AdamsTree( $T_1, \dots, T_k$ )  
  if  $T_1$  contains one leaf  
    return  $T_1$   
  Construct  $\pi(T) : \Pi(\pi(T_1), \dots, \pi(T_k))$   
  For each block  $B$  in  $\pi(T)$   
    AdamsTree( $T_1|B, \dots, T_k|B$ )  
  Attach the root of these trees to a new node  $v$   
  return this tree
```

$T_i | X$ means a restriction. It's defined by: For every cluster A in T_i the output will be $A \cap X$.

Definition

π_1, \dots, π_n is a partition of the set of taxa.
 Π is the product of π_1, \dots, π_n

The product of these partitions is the partition for which two taxa a and b are in the block iff they are in the same block for each π_1, \dots, π_n

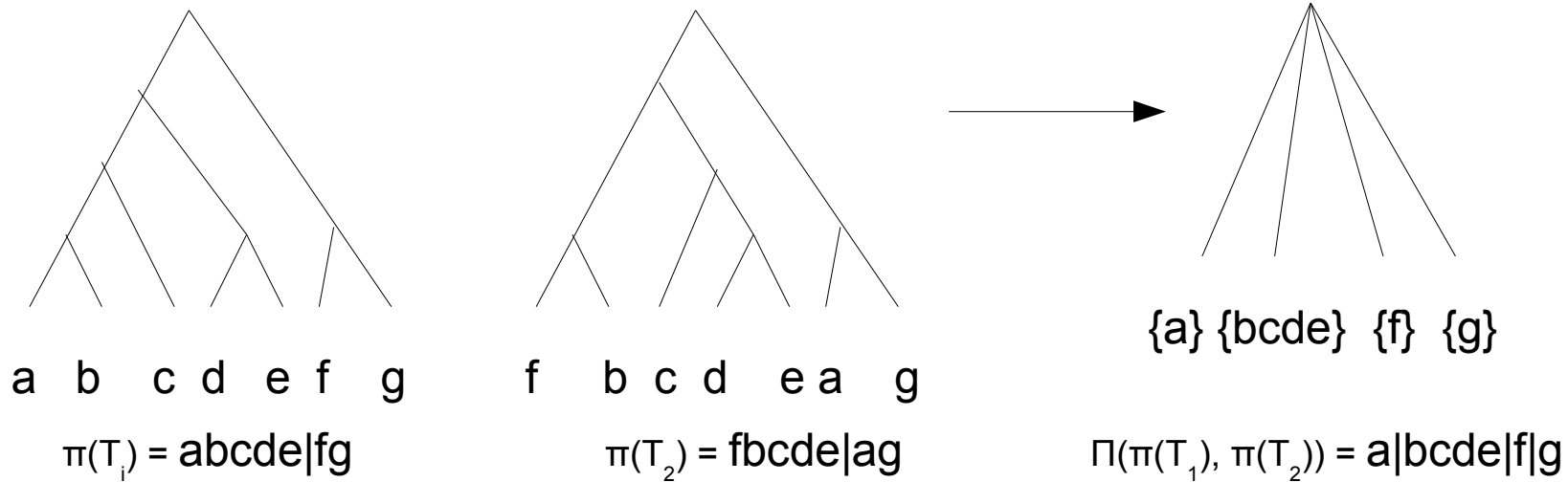
E.g. $ab|c, ac|b$ gives $a|b|c$

Definition

The maximal clusters of a tree T_i are the largest proper clusters in T_i .

The maximal cluster partition for T_i is the partition $\pi(T_i)$ of the set of taxa with blocks equal to the maximal clusters of T_i

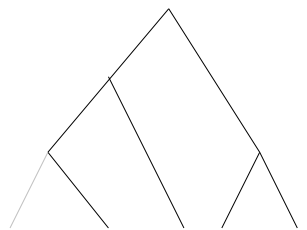
2.1 Adams consensus example



AdamsTree($T_1|_{\{b,c,d,e\}}, T_2|_{\{b,c,d,e\}}$)

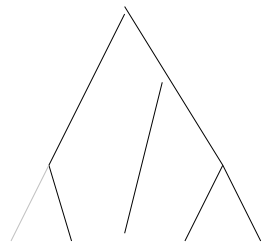
$A = T_1|_{\{b,c,d,e\}}$

$B = T_2|_{\{b,c,d,e\}}$



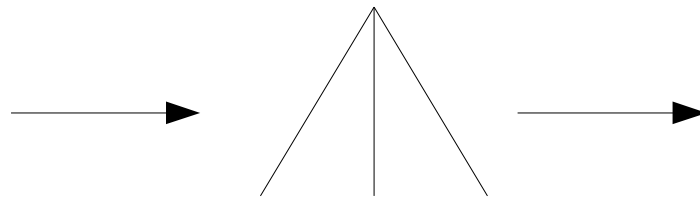
b c d e

$\pi(A) = bc|de$



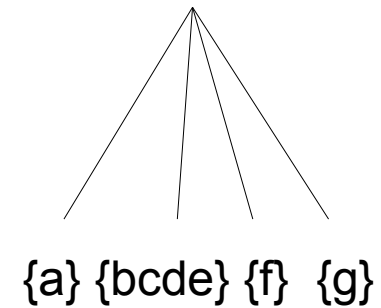
b c d e

$\pi(B) = b|cde$

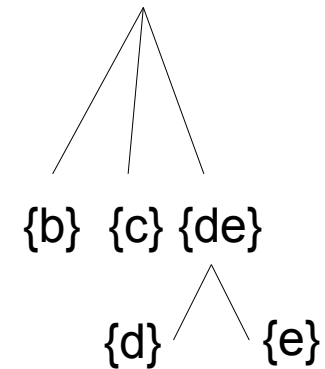


{b} {c} {de}

$\Pi(\pi(A), \pi(B)) = b|c|de$



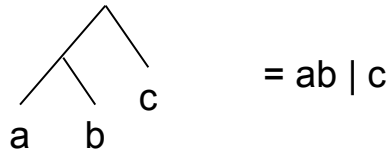
{a} {bcde} {f} {g}



{b} {c} {de}

{d} {e}

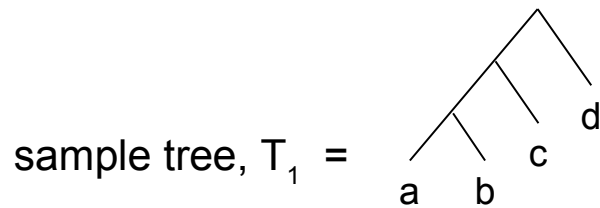
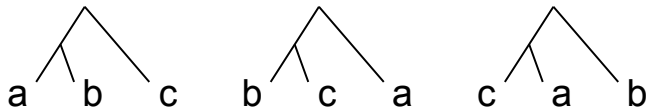
3 Methods based on sub trees



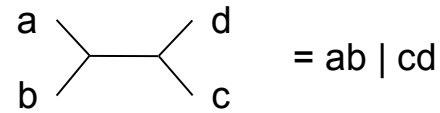
a rooted tree, T , contains a rooted triple, $ab|c$, if the least common ancestor, $\text{lca}(a,b)$ descends $\text{lca}(a,b,c)$.

$r(T)$ = the set of rooted triples in T .

rooted triplets has 3 configurations:



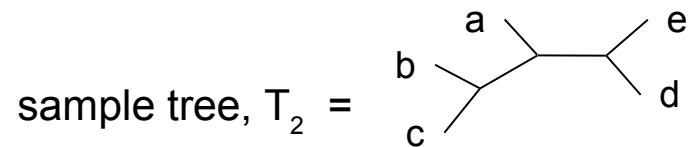
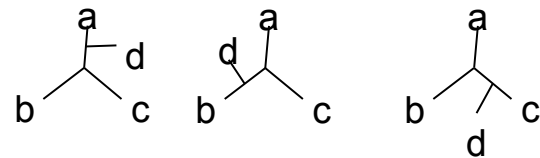
$r(T_1) = \{ ab|c, ab|d, ac|d, bc|d \}$.



an unrooted tree, T , contains a quartet $ab|cd$, if the path from a to b in T does not intersect the path from c to d .

$q(T)$ = the set of quartets in T .

binary quartets has 3 configurations:



$q(T_2) = \{ ab|de, bc|de, ac|de \}$.

3.1 Local consensus tree

given a collection of rooted trees, $T = \{T_1, \dots, T_N\}$

input:

$R = \bigcap_{i=1..N} r(T_i)$ = the set of rooted triples appearing in all trees of T .

L = set of leafs to compute.

precondition:

the set R restricted by L is compatible!

return:

$T_{\text{consensus}} = \text{OneTree}(R, L)$

Pseudo code, Procedure OneTree(R,S)

1. If $n = 1$ then return a single vertex labelled by x_1 .
 2. If $n = 2$ then return a tree with two leaves labelled x_1 and x_2 .
 3. Otherwise, construct $[R, S]$ as described.
 4. If $[R, S]$ has only one component then return 'No Tree'.
 5. For each component S_i of $[R, S]$ do
 6. If OneTree(R, S_i) returns a tree then call it T_i else return 'No Tree'.
 7. end(for)
 8. Construct a new tree T by connecting the roots of the trees T_i to a new root r .
 9. return T .
- end.

property of solution:

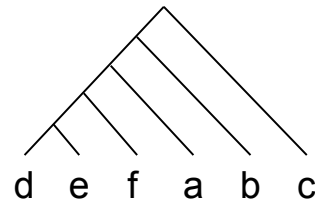
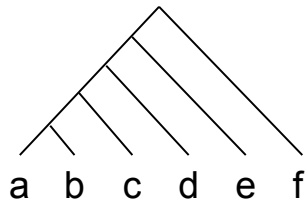
$R \subseteq r(T_{\text{consensus}})$, all common input triple are contained.

3.2 Local consensus tree example

Problem:

Inp. $T_1 = (((((a, b), c), d), e), f)$

Inp. $T_2 = (((((d, e), f), a), b), c)$



T_1 :

Input leafs, $L = \{a, b\}$
Input triplets, $R = \{ab|c, de|f\}$

Restrict R by $L = \emptyset$
 \Rightarrow nothing to connect.

Return: $ab|local_root$

T_2 :

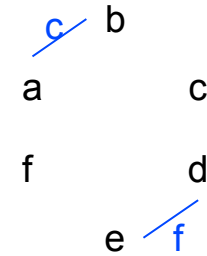
...

Return: $ed|local_root$

$T_{consensus}$:

Input leafs, $L = \{a, b, c, d, e, f\}$
Input triplets, $R = \{ab|c, de|f\}$

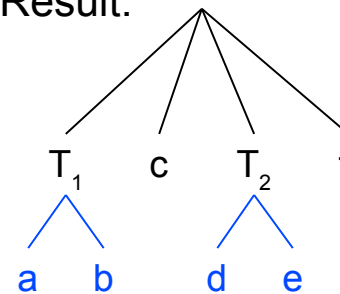
Restrict R by $L = R$
and connect L using R



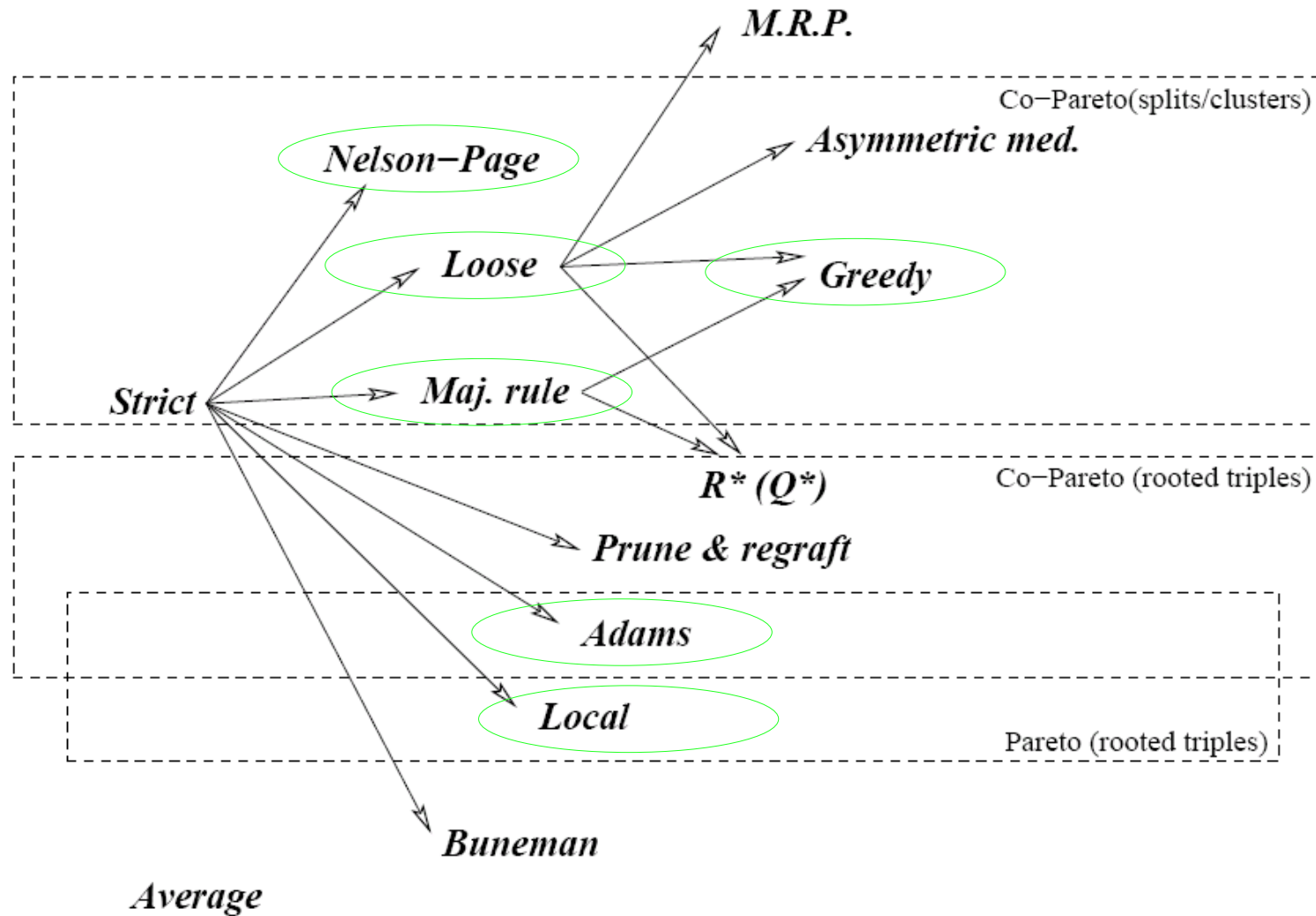
Components:

$T_1: \{a, b\}, \{c\}, T_2: \{d, e\}, \{f\}$.

Result:



4 Classification of consensus tree methods



green circle = included in this presentation

Figure: A classification of consensus tree methods. [Bryant2003]

5 Consensus methods for trees with different taxa set.

General idea:

- Given tree A with taxa {a,b,c}
- Given tree B with taxa {e,f}
- We will construct a tree with taxa {a,b,c,e,f}

From quartets to phylogenetic trees

- Problems in constructing a phylogenetic tree:
 - Few taxa – much data – Precise tree but with few taxa (called Taxonomic sampling)
 - Many taxa – less data – Often a biased result
- Distance- and character based methods has these problems – We need something better.

The Four Taxon approach

Idea: take a subset of taxa on size 4, take all protein sequences known for the subset of taxa, construct quartets. Avoids both problems.

Output: Unrooted tree (can be rooted using an outgroup)

5.1 Problem Description

- C_j - positive weight (a real number in the interval $[0;1]$) of the j 'th quartet.
- We call it the confidence value and it's defined as:
 - The strength of the phylogenetic signal.
 - Size of the sequence population.
- We can score a tree T built of the set of quartets Q

$$score_Q(T) = \sum_{S \in S} C_S + \frac{1}{3} \sum_{U \in U} C_U$$

$S \subseteq Q, U \subseteq Q$

- The set S contains satisfied quartets (tree topology = quartet topology)
- The set U contains unresolved quartets (star topology)
- The sets are determined by. Let's consider a quartet $ab|cd$
 - Remove all nodes but a, b, c, d .
 - Adjacent edges is deleted
 - Internal nodes with degree 2 is deleted (connect adjacent nodes)
 - Then we examine the topology.
- We want to maximize the score.
- The problem is NP-hard. (Reduction from MAX-CUT)

Algorithms

- The exact algorithm (mild exponential running time)
- Quartets puzzling – the greedy heuristic
- The geometric heuristic

5.2 Quartet puzzling

Quartet Puzzling

input: set of quartets, Q , on total set of taxa or species, S .

repeat many times:

permute S

execute Puzzling Step (Q, S)

add tree to collection of results

return: the majority consensus tree

Puzzling Step ($Q, S = \{a, b, c, d, e, f, g, \dots\}$)

select the quartet topology of $\{a, b, c, d\}$ as anchor

for ($s = e, f, g, \dots$):

reset edge counters

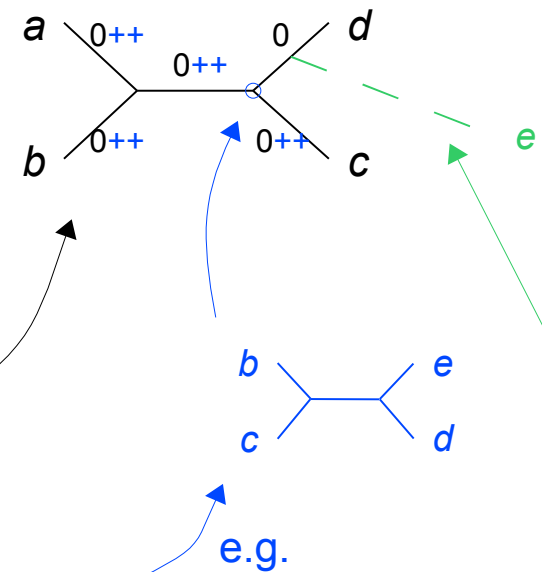
for every quartet, $q = i | j | k | s$, in Q , where $(i, j, \& k) < s$:

locate the node, O , that intersects the paths between i, j & k

increment every counter in every sub tree from O in conflict with q

branch from edge with minimum count and add new leaf, s

return resulting tree



5.3 The Geometric Heuristic

- The problem can be addressed geometrically, and solved by using SDP (Semi definite programming).
- SDP gives an approximation, with any desirable precision.

Idea:

- Use a unit sphere in \mathbb{R}^n where n is the number of taxa.
- For every quartet $ab|cd$, place a and b close to each other but a and d far from each other. a, b, c, d is placed on the boundary of the unit sphere.
- We can now formulate the semi definite programming problem:

Maximize

$$\sum_{1 \leq j \leq k} C_j (\langle a_j, b_j \rangle + \langle c_j, d_j \rangle) - 0.5 \sum_{1 \leq j \leq k} C_j (\langle a_j, c_j \rangle + \langle a_j, d_j \rangle + \langle b_j, c_j \rangle + \langle b_j, d_j \rangle)$$

Subject to

$$\langle v_i, v_i \rangle = 1 \quad (1 \leq i \leq n)$$

- Max = 4C: a and b is placed at the same point and c and d is placed at the antipodean point. (The diametrically opposite point).
- Min = -4C: a and c is placed at the same point, b and d at the antipodean point.

5.4 The Geometric Heuristic

Improvements:

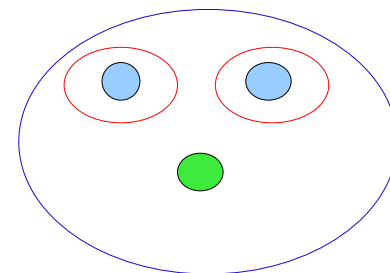
- Threshold for confidence level – quartets with low confidence level may be built from inconsistent data
- To avoid points to be placed at the same point, we want a small distance between the points. We can add this constraint to the SDP problem:

$$\langle v_i, v_j \rangle \leq 1 - \epsilon \quad (1 \leq i < j \leq n)$$

ϵ should be relatively small (e.g. 0.25), We'll obtain a better score. .

Geometrical Clustering

- After we have solved the SDP problem, we've each quartets placed (as points) on the unit sphere. We want to join them to get a tree.
- This is done in this way:
 - Initialization: n clusters, each contains a single point
 - At each step, we decrease the number of clusters. The procedure terminates when $|\text{clusters}| = 1$
 - At each step, we remove 2 clusters and add 1
 - Selection is done by calculating pairwise euclidean distance
 - The point associated with the new cluster is the center of mass of the points of the removed clusters (i.e. $|\text{taxas}|$)



References

[Bryant2003]

David Bryant, "A Classification of Consensus Methods for Phylogenetics", Bioconsensus, Proc. of Tutorial and Workshop on Bioconsensus, II DIMACS-AMS, (2003) 55-66.

[Chor1998]

Benny Chor, "From Quartets to Phylogenetic Trees", B. Rovin (Ed.): SOFSEM' 98: Theory and Practice of Informatics, LNCS 1521, pp. 36-53, 1998.