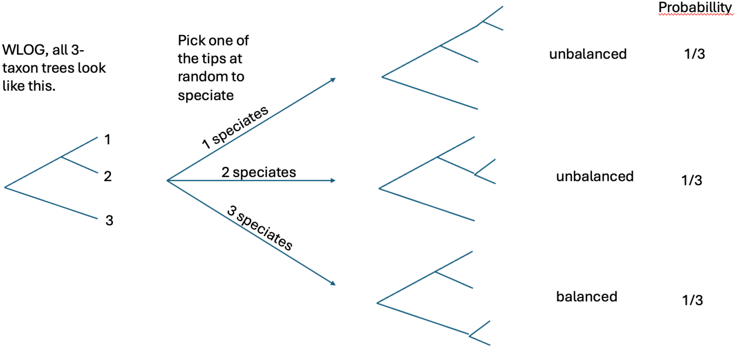
An interesting question is what are some probability models for trees? Is each tree equally likely? What if we consider tree shapes where we remove the labels. Is each tree shape equally likely?

We saw on a previous slide that the number of rooted, binary tree topologies on 4 taxa is 15. Of these 15 trees, 12 are unbalanced, and 3 are balanced. So if we picked a tree at random from this set of 15, there would be $3/15 = 1/5 = 0.2$ probability that the tree was balanced (two descendants on each side of the root).

Is this the same probability that get from a birth process such as generated by `TreeSim` in R?



WLOG, all 3-taxon trees look like this.

1

2

3

Pick one of the tips at random to speciate

1 speciates

2 speciates

3 speciates

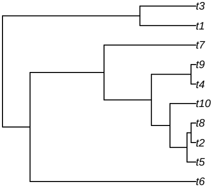
	Probabillity
unbalanced	1/3
unbalanced	1/3
balanced	1/3

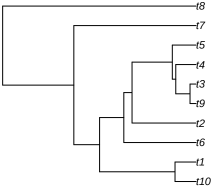
In the case of 4 taxa, we have the birth process giving a $1/3$ probability to the balanced tree shape, and the uniform model (also called PDA for Proportional to Distinguishable Arrangements) giving a $1/5$ probability to the balanced tree shape. This has interesting applications for Bayesian statistics inference on trees. If we want a prior distribution on trees, should we use the PDA model or the birth model? We see for 4 taxa that the PDA model tends to favor less balanced trees. This holds true for larger trees as well.

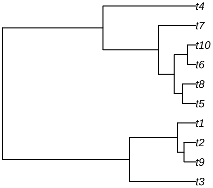
There is a literature on measuring tree shape in phylogenetics. In addition to priors for Bayesian analyses, we might be interested in whether empirically estimated phylogenies resemble the birth (Yule) model or the PDA model more.

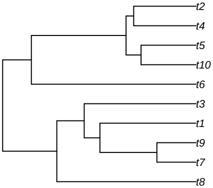
```
library(treestats)
set.seed(2025)
#rtree(10) also generates 10-taxon trees
x <- sim.bd.taxa(10,4,1,0)
colless(x[[1]])
[1] 22
colless(x[[2]])
[1] 29
colless(x[[3]])
[1] 12
colless(x[[4]])
[1] 9
```

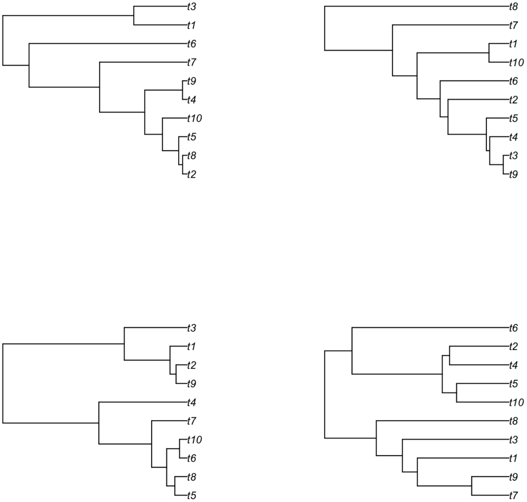
```
plot.phylo(x[[1]])
plot.phylo(x[[2]])
plot.phylo(x[[3]])
plot.phylo(x[[4]])
plot.phylo(ladderize(x[[1]]))
plot.phylo(ladderize(x[[2]]))
plot.phylo(ladderize(x[[3]]))
plot.phylo(ladderize(x[[4]]))
```


t3
t1
t7
t9
t4
t10
t8
t2
t5
t6


t8
t7
t5
t4
t3
t9
t2
t6
t1
t10

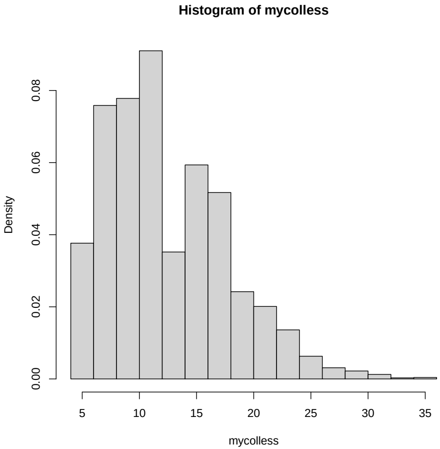

t4
t7
t10
t6
t8
t5
t1
t2
t9
t3


t2
t4
t5
t10
t6
t3
t1
t9
t7
t8


t3
t1
t6
t7
t9
t4
t10
t5
t8
t2
t8
t7
t1
t10
t6
t2
t5
t4
t3
t9
t3
t1
t2
t9
t4
t7
t10
t6
t8
t5
t6
t2
t4
t5
t10
t8
t3
t1
t9
t7

Let's try doing a simulation to get something like the distribution of the Colless statistic. Imagine we have a null hypothesis that the tree is generated under a Yule model and the alternative is that it is PDA (or that it is just not Yule). We have a 10-taxon tree with a Colless statistic of 31. Is this plausible under the Yule model?

```
> x <- sim.bd.taxa(10,10000,1,0)
for(>
> mycolless <- rep(0,10000)
> for(i in 1:10000) {
+ mycolless[i] <- colless(x[[i]])
+ }
> hist(mycolless)
> mean(mycolless>=31)
[1] 0.0039 #one-sided p-value
```


Histogram of mycolless
Density
0.00
0.02
0.04
0.06
0.08
5
10
15
20
25
30
35
mycolless

The Colless statistic is

$$C = \sum_i |L_i - R_i|$$

where L_i and R_i are the number of leaves descended from the left and right side of vertex i.

Other indices of tree balance include the Sackin index (sum of the number of edges from the root to each tip, and the number of cherries (two-taxon clades)

```
dim(cherries(x[[1]]))
```

```
sackin(x[[1]])
```

```
> tree_str1 <- "(((A,B),C),(D,(E,F)));"
> tree_str2 <- "(((A,B),(C,D)),(E,F));"
> tree1 <- read.tree(text=tree_str1)
> tree2 <- read.tree(text=tree_str2)
> colless(tree1)
[1] 2
> colless(tree2)
[1] 2
> sackin(tree1)
[1] 16
> sackin(tree2)
[1] 16
> dim(cherries(tree1))[1]
[1] 2
> dim(cherries(tree2))[1]
[1] 3
> cherries(tree1)
   [,1] [,2]
9  "A"  "B"
11 "E"  "F"
```

Homework 1 (two problems).

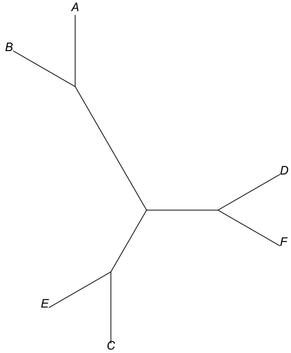
1.. Write a simulation to estimate the probability under the Yule (birth) model, that for 10-taxon trees on species $A, B, C, D, E, F, G, H, I, J$, $\{A, B\}$ is a clade. To do this, generate many trees using `TreeSim`, for each tree, determine whether `(A,B)` or `(B,A)` is a cherry in the tree. You can either use the `cherries` function to do this or use string operations in R. For example,

```
x <- "((A,B),C)"
grepl("(A,B)",x)
[1] TRUE
```

Homework 1 continued

.

2. For the unrooted tree topology below, draw (by hand) all rooted tree topologies with this unrooted topology.



The newick representation of an unrooted tree can be obtained using the regular newick notation with a 3-way split at the “root”, such as `((A,B),C,D)` and `((A,B),D,(C,E)`

For unrooted trees, there is an analogous concept to clades which are called **splits** or **bipartitions**. The idea of a split is that if you remove an edge, you've split the graph into two connected components, and this partitions the taxa into sets, or splits.

For the 4-taxon tree on the previous slide, the non-trivial split is $ab|cd$. For the 5taxon tree, the splits are $ab|cde$ and $abd|ce$.

Rooted trees are completely described by their clades. Unrooted trees are completely described by their splits. The list of splits that make up a tree is also called a **split encoding** of a tree.

A set of clades is **compatible** if there is a tree (possibly not fully resolved) that has those clades on them. Two clades (i.e., subsets of taxa) $\mathcal{C}_1$ and $\mathcal{C}_2$, are compatible if

- $\mathcal{C}_1 \subset \mathcal{C}_2$ or $\mathcal{C}_2 \subset \mathcal{C}_1$ (one is s subset of the other), or
- $C_1 \cap C_2 = \emptyset$ (they have no clades in common)

Corollary 2.11 in the book states that a set of subsets of taxa is compatible if and only if every pair of subsets is compatible. In other words, pairwise compatibility implies compatibility of the entire set.

Similarly for splits, a set of bipartitions or splits is compatible if there is an unrooted tree with all splits in the set. (Theorem 2.20 in the book).

Let two bipartitions be of the form $X_1|X_2$ and $Y_1|Y_2$ (i.e., X_1, X_2, Y_1, and Y_2 are subsets of the complete set of taxa on the tree). Then the bipartitions are compatible if and only if at least one of the the pairwise intersections $X_i \cap Y_j$ are empty.

Given a list of proposed splits, this gives a way of checking for compatible of the entire list by checking two at a time.

Instead of going through the proof (which is in the book), we'll go through an example to see how this works. The idea for the proof, though, is that two bipartitions will be compatible if the tree can be rooted in such a way that their clades are compatible.

Let the taxon set be $\{a, b, c, d, e, f, g\}$, so 7 taxa. We'll consider the following bipartitions

- $abc|defg$
- $ab|cdefg$
- $abcde|fg$

Is the previous list of clades compatible?

Let the previous list of splits be represented by $X_1|X_2$, $Y_1|Y_2$, and $Z_1|Z_2$.

- $X_1 = \{abc\}$
- $X_2 = \{defg\}$
- $Y_1 = \{ab\}$
- $Y_2 = \{cdefg\}$
- $Z_1 = \{abcde\}$
- $Z_2 = \{fg\}$

Note that $X_2 \cap Y_1 = \emptyset$ so that the first two splits are compatible. Also, $X_2 \cap Y_1 = \emptyset$, so the first and second splits are comaptible. Finally, $X_1 \cap Z_2 = \emptyset$, so the first and third splits are compatible. Since all three pairs are compatible, the entire set is compatible.

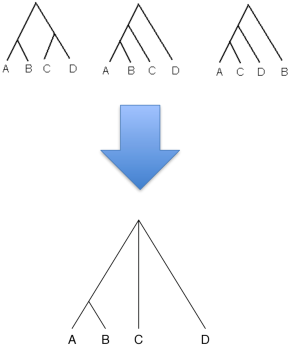
Given either a set of compatible clades, or a set of compatible splits, it is possible to construct a tree algorithmically. In practice what happens from data is we are often given a set of clades or splits that are not compatible, but algorithms can be found to try to build a tree anyway by minimizing the amount of conflict in the input data.

Consensus trees

A consensus tree is a tree that takes other trees as input, and constructs an output tree. Sometimes the input trees are not fully resolved, and the output tree might be more resolved than the input trees. The reverse can also happen if the input trees have a lot of conflicts, where the output is less resolved than the input trees.

There are many types of algorithms for consensus trees, but one approach is to take input trees, represent them by their clades or splits, and then construct a tree from the list of clades or splits. This is more straightforward if the clades or splits are compatible, and less straightforward if they are not compatible.

Consensus methods



Other methods exist for constructing consensus trees. For example, instead of clades or splits, we might use **rooted triples** (rooted) or **quartets** (unrooted).

A rooted triple, is the tree obtained from a larger tree when only a subset of three taxa is considered. For example, for the tree (((a,b),c),d) the rooted triples are

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

For the tree (((a,b),c),(d,e)), the unrooted quartets are

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

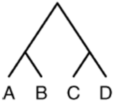
Why are consensus trees used?

There are different reasons. Historically, sometimes two trees were inferred from different data sources, say one morphological, and one DNA-based, or one based on miDNA and one based on nuclear DNA (one gene each used to be common). When there were conflicting trees, a consensus tree could be used to summarize what the two trees had in common, and leave the rest unresolved. Or two trees might be unresolved in different places, and combining could lead to a more resolved tree.

Majority rule—consensus tree has all clades that were observed in > 50% of trees.

Greedy—sort clades by their proportions. Accept the most frequently observed clades one at a time that are compatible with already accepted clades. Do this until you have a fully resolved tree.

R*—for each set of 3 taxa, find the most commonly occurring triple e.g., (AB)C, (AC)B or (BC)A. Build the tree from the most commonly occurring triple.



(AB)D, (CD)B are
two rooted triples

ASTRAL—a median tree. Find the tree that minimizes the sum of distances to input gene trees. For ASTRAL, a quartet distance is used.

for more details, the majority-rule tree includes all clades (and only clades) that occur in more than 50% of the input trees. These clades are necessarily compatible but the resulting tree might not be resolved.

For the greedy consensus tree, further resolve the majority-rule tree by accepting clades one at a time that have the highest frequency and are compatible with previously accepted clades. In a tie, choose a clade arbitrarily or at random.

The rooted triple algorithm is the most complicated. Call rooted triples that occur more frequently than any conflicting triple **uniquely favored**. These do not necessarily occur in more than 50% of input trees, and are not necessarily compatible (cannot necessarily all occur on the same tree). There also can be ties. Get a list of uniquely favored triples.

Next, the most resolved tree that contains only uniquely favored triples is constructed. Use the following rules. Let S be the complete set of taxa.

- Clades of size 1 and n are automatically included.
- $\mathcal{A}$ is a clade exactly when for each pair of taxa $A_i, A_j \in Awithi \neq j$, every taxa $Z \in S \setminus A, (A_i, A_j)Z$ is uniquely favored.

To make this more concrete for 4 taxa, A, B, C, D, $\{A, B\}$ is a clade if and only if $(AB)C$ and $(AB)D$ are uniquely favored. $\{A, B, C\}$ is a clade if and only if $(AB)D$, $(AC)D$, and $(BC)D$ are uniquely favored.

Example: suppose the input gene trees are $(((AB)C)D), (((AD)C)B), (((BC)A)D), and (((CD)A)B)$. Find the Majority-rule, Greedy, and R* trees.

For Majority rule and greedy, let's make a list of clades.

Clade	count
AB	1
AD	1
BC	1
CD	1
ABC	2
ACD	2

Since no clade occurs more than 50% of the time, the majority-rule consensus tree is completely unresolved. The greedy tree accepts either $\{ABC\}$ or $\{ACD\}$ as a clade first, then resolves it arbitrarily. Thus the greedy consensus tree could be either $(((AB), C), D), (((BC)A)D), (((AD)C)B)$, or $(((CD)A)B)$, exactly one of the input trees.

For the R* approach, make a list of the rooted triples. The number of rooted triples is $\binom{4}{3} = 4$ per tree, so a total of 16.

Triple	count
$(AB)C$	1
$(AB)D$	2
$(AC)D$	2
$(BC)D$	2
$(BC)A$	1
$(AD)C$	1
$(AD)B$	2
$(AC)B$	2
$(CD)B$	2
$(CD)A$	1

For $\{A, B, C\}$, $(AC)B$ is uniquely favored since it shows up twice and conflicting triples show up once each. For $\{A, C, D\}$, (AC)D is uniquely favored. Therefore since (AC)D and (AC)B both are unqiuely favored, $\{AC\}$ is a clade on the consensus tree. For $\{A, B, D\}$, $(A, B)D$ and $(AD)B$ each show up twice, so neither is uniquely favored. Therefore the only clade in the R* consensus tree is (AC), and the resulting tree is unresolved as ((A,C),B,D).