coalescent 模拟的主要目的是判断观察到的基因树与物种树之间的冲突是否可以用不完全谱系分选来解释。其主要思路是，先获得一棵物种树 (物种树需要由ASTRAL之类的软件依据coalescent模型生成)，再依据这棵物种树，完全根据coalescent模型 (coalescent模型即为不完全谱系分选的数学模型) 去模拟多棵基因树。最后计算这些模拟出的基因树和物种树之间的树形距离的分布以及计算实际测序得到的基因树与物种树之间的树形距离的分布。如果这两个分布大致相同，那么我们认为，不完全谱系分选是可以很大程度上解释基因树和物种树之间的冲突的。而如果分布形状差别较大，那么我们就认为不完全谱系分选不足以解释基因树和物种树之间的冲突。

但其实这种分析只是一个定性分析，只可获得一个直观的判断，无法具体的分析出coalescent模型能解释多少比例的冲突。2019年science的一篇文章 “Genomic architecture and introgression shape abutterfly radiation” 中发表了一个名为 “QuIBL” 的软件，该软件可以分别计算出不完全谱系分选与基因流时间具体能解释多少比例冲突，达到定量分析的目的。

本 coalescent 模拟流程主要修改自文章 “Incomplete lineage sorting rather than hybridization explains the inconsistent phylogeny of the wisent” 中所使用的方法。该文章作者将过程中使用的脚本都上传到了 github 中：https://github.com/wk8910/ILS\_simulation。 不过文章作者最后呈现结果是以一个小提琴图呈现的，我们认为普通的柱状的频率分布图 (类似于文章 Disentangling Sources of Gene Tree Discordance in Phylogenomic Data Sets: Testing Ancient Hybridizations in Amaranthaceae s.l中的figure 5a) 更直观一些。另外，这篇文章中的流程，其实相对来说是一个比较简略的流程。一般情况下在模拟基因树的时，需要指定各个古代类群的种群大小，不同的种群大小会显著影响不完全谱系分选发生的概率。而实际研究中一般很难获取古代类群的种群大小，所以部分文章会设置一个最小的种群大小以及一个最大的种群大小，并分别进行计算，但此方法比较繁琐。这篇文章的流程只是将种群大小设置为了一个固定的数值。

依赖软件安装：(本流程所有依赖软件及脚本均仅在linux环境下经过测试)

1. ete3:

ete3是一个Python模块，该模块用来处理系统发育树树形相关的分析。但该模块设计较复杂，需要依赖多个其他模块，所以尽量使用其网站<http://etetoolkit.org/download/> 中推荐的安装方法，将该模块安装到一个独立的conda子环境中，避免其依赖包与在计算机中已安装的包在版本上发生冲突。后面需要安装的其他模块也可安装到这个子环境中。

1. dendropy:

dendropy也是一个Python模块，他主要的作用是计算基因树和物种树之间的距离

1. pandas、matplotlib、seaborn：

其他的一些Python常用模块，用来统计树形，绘制最后的频率分布图等等

1. phybase：

phybase是一个R模块，根据物种树模拟基因树时需要使用这一模块。

具体步骤：

1. 从基因树中提取子树 (subtree)，并且重新置根

这一步操作的主要原因有两个，首先是在我们输入的每一棵基因树可能都是有missing taxa的，而最后我们要比较基因树与物种树之间的拓扑结构距离，这就需要基因树和物种树之间有完全相同的物种名称。所以我们要通过提取子树的方法保证我们分析的所有树都包含同样的物种名称。另外，如果我们的树中包含太多的物种，那么最后基因树与物种树之间的拓扑距离就可能非常大，而拓扑距离的离散程度也就可能非常大，这样就需要非常多的基因树才可能获得一个置信程度高的统计结果。而一般来说，我们实际能获得的基因树是有限的。所以尽量仅选取各个分支中的代表物种，提取子树进行分析。

1. 进入文件夹“0\_extract\_subtree”。
2. 输入文件1：“input\_gene.trees”，该文件中每行代表一棵基因树，基因树应为newick格式。
3. 输入文件2：“subtree\_species.txt”，该文件中的每一行代表你想提取的基因树子树的一个物种名称 (包含外类群)，注意这里的物种名称应与基因树中实际的物种名称严格对应。
4. 两个输入文件都拷入文件夹“0\_extract\_subtree”后，使用以下命令运行脚本：

“python extract\_subtrees.py -i input\_gene.trees -n subtree\_species.txt -o Hydrastis\_canadensis\_2.fasta.transdecoder.pep”

其中，参数-i为输入文件1的文件名称，参数-n为输入文件2的文件名称，参数-o为置根时所用的外类群名称。

1. 输出文件：“output\_subtree.trees”，该文件中每一行代表一棵提取出的子树，但注意这里的子树的数量可能会比输入的基因树少的多，这是由于当一棵输入基因树没有包含所有子树中的物种名称时，该基因树将被删除。

1. 使用刚刚提取出的基因子树构建一棵ASTRAL物种树

上一步中我们已经获得基因树的子树，这一步中将使用ASTRAL软件构建一棵物种树，并将ASTRAL的输出树转化为一棵“超度量”的树 (coalescent模拟使用的R模块要求输入的物种树为“超度量”的树)。

1. 进入文件夹“1\_astral”。
2. 输入文件1：“output\_subtree.trees”，该文件为上一步骤的输出文件。
3. 将输入文件拷入文件夹“0\_extract\_subtree”后，使用以下命令运行脚本：

“python run.py”。

1. 输出文件：“astral.tre.regular.tre”，该文件为ASTRAL软件输出的物种树，且已经转化为“超度量”的树。
2. 使用上一步生成的ASTRAL物种树依据coalescent模型来模拟基因树

上一步中我们已经获得了超度量的物种树，这一步中将使用“phybase”模块依据coalescent模型模拟出该物种树对应的基因树。

1. 进入文件夹“2\_simulations”。
2. 输入文件1：“astral.tre.regular.tre”，该文件为上一步骤的输出文件。
3. 将输入文件拷入文件夹“2\_simulations”后，使用以下命令运行脚本：

“python simulate.py -n 10000 -o Hydrastis\_canadensis\_2.fasta.transdecoder.pep”。

其中，参数-n为需要模拟出的基因树数量，建议该数量不要小于10000，参数-o为给模拟出的基因树置根时所用的外类群名称。

1. 输出文件：“astral.simulate.reroot.tree”，该文件为模拟出的基因树，文件中的每一行代表一棵基因树。
2. 比较实际的基因树与模拟的基因树拓扑结构，并绘图

我们在步骤(1)中我们获得了实际的基因树 (其实是基因树的子树)，在步骤(2)中我们获得了物种树，在步骤(3)中我们获得了模拟出的基因树。在本步骤中，我们要先计算实际的基因树与物种树之间的拓扑结构距离，之后再计算模拟出的基因树与物种树之间的拓扑结构距离。之后将这些距离转化为距离分布直方图，并对比两个分布是否相似。

1. 进入文件夹“3\_compare”。
2. 输入文件1：“output\_subtree.trees”，该文件步骤(1)的输出文件。
3. 输入文件2：“astral.tre.regular.tre”，该文件为步骤(2)的输出文件。
4. 输入文件3：“astral.simulate.reroot.tree”，该文件为步骤(3)的输出文件。
5. 将三个输入文件均拷入文件夹“3\_compare”后，使用以下命令运行脚本：

“python compare\_trees\_distance.py”。

1. 输出文件1：“compare.png”，该文件为脚本最终输出的频率分布图，横坐标中的数值表示基因树(实际的与模拟出的)与物种树之间拓扑结构冲突数目为该数量的频率。
2. 输出文件2：“result.tsv”，该文件为输出文件1的原始数据，可以使用该文件重新绘制频率分布图。