SNaQ 2.0

Nathan Kolbow, Sungsik Kong & Claudia Solís-Lemus

October 13, 2023

The goal of this study is to introduce a new software, SNaQ 2.0, which is an updated version of original SNaQ with an improvement in computational cost. In this study, we want to emphasize that SNaQ 2.0 is faster than the original SNaQ with their accuracy being equal.

1 Pipeline

We must start thinking about how to use HTC for this project.

1.1 Species network generation

We decided to create 9 different network topologies that varies in size and complexity, controlled by the number of tips n and reticulations r in the network. We set $n \in \{10, 25, 50\}$ and $r \in \{1, 3, 5\}$. We generate the species networks semi-manually. Current plan is to produce tree topology under some model (i.e., Yule process) and sequentially as reticulations onto the topology. But there might be a better way. For each species nework, we make three versions that vary in terms of branch lengths. We do this to vary the amount of ILS in the data.

1.2 Gene tree and sequence generation

For each species network, we use `PhyloCoalSimulations` to generate g gene trees. We set $g \in \{50, 100, 500, 1000, 3000\}$. Then, we use $seq - gen$ to generate sequence alignment for each of the gene trees We have to decide sequence length and some other parameters like which evolutionary model we will use.

1.3 Gene tree estimation

For each simulated sequence alignment, we estimate a gene tree using `iq-tree 2`. We then compute gene tree estimation error (GTEE) as well.

1.4 Network estimation

Using the set of gene trees, we estimate species network using SNaQ 1.0 and SNaQ 2.0. We assume the true number of reticulations in the final network is known. Based on Tyler's Notion, it seems like **multithreading** the psuedolikelihood calculation and probQR are two new features in SNaQ 2.0 that improves computational speed. Obviously, we would like to test the performance of these. It seems like speed-up varies between n and the **depth** of the network, but nothing on the effect of r was tested. We might want to test method in sing number of cores (possibly 10?), different numbers of **threads** ($\in \{4, 8, 12, 16\}$?) and **probQR** (possibly $\in \{0.25, 0.5, 0.75\}$).

Question: How can we know the depth of the tree? Can we simply use the coalescent unit from the root to the tip? We will have tree different depths (different number of ILS).

Tyler uses core and threads indiscriminately, just want to make sure that they are the same thing. The reason I am asking this is because Tyler parallelized the pseudolikelihood computation using Threads.@thread and each run is parallelized using pmap. When opening Julia, the number of threads can be specified using the flag -t and the number of processors can be specified using -p. I am pretty sure the two are different things.

Therefore, we might want to something like this:

```
procs=[4, 8, 12, 16]
probQR=[0, 1/3, 2/3, 1]
propQ=[1, 0.9, 0.7]
for nproc in procs
for pr in probQR
julia -t nproc -p nproc
net1=SNaQ 1.0 # Vanilla vanilla
net1=SNaQ 2.0 # This parallelizes PL computation plus probQR
end
end
```

1.5 Analysis

To analysize the accuracy, we compare each of the estimated species network with the true topology. The network dissimilarity is measured using hardwired cluster distance.

To analyze the computational cost, we can use the time taken reported in the output of SNaQ. Alternatively, may be we can use CPU time reported by the cluster.

We use `ggplot2` to plot the output.

1.6 Empirical datasets

First option is to replicate the swordflish network reported in the SNaQ 1.0 paper.

We might want to use another empirical dataset. Some preferences are the dataset of the biological system where hybridization is well studied, contains estimated set of gene trees, and quite reliable.

1.7 Submission

Currently, software note on Bioinformatics or Methods in Ecology and Evolution seem to be feasible options. We shoot for late December to do this.