BioNetwork Research: WindowReps 2019 Winter Report

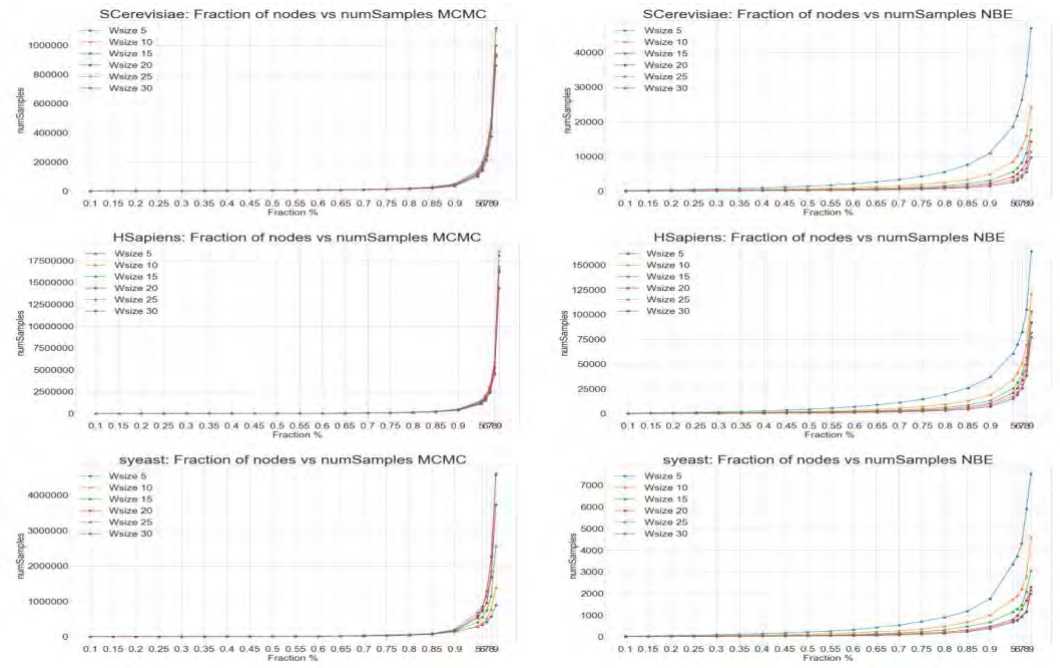
Henry Ye

Continued from last quarter where I experimented different window representatives searching methods and discussed the tradeoffs between Uniqueness and Stability, this quarter, I began implementing window representatives into “BLANT”.

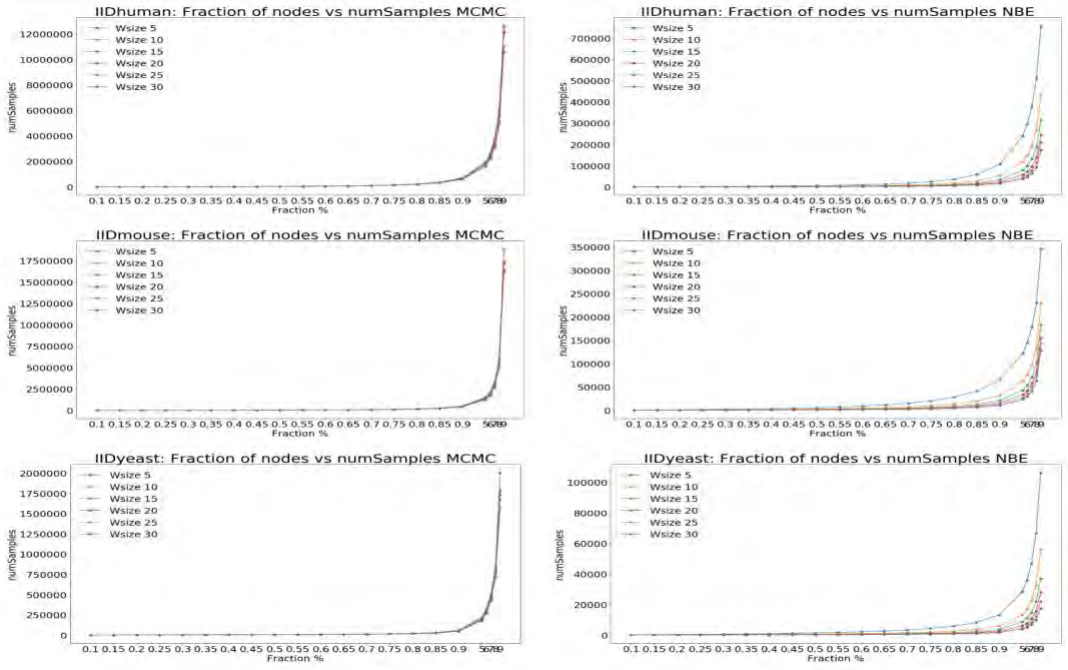
At the beginning of this quarter, we focused on the uniqueness property of windowReps since we wanted to construct network databases. Therefore, “Least Frequent” searching method, which can maximize uniqueness of windowReps, was used. However, “Least Frequent” requires storing and iterating through all k-node graphlets twice to determine windowReps, resulting in a very long runtime (roughly 1 minute for sampling 1,000 20-node windows with windowRep k=7). In order to reduce the runtime, I looked into the gprofile outputs, and then modified graphlet canonical integer conversion function; added dynamic array to libwayne; found and implemented ESU (a DFS like algorithm which can greatly reduce the runtime when the $\binom{W}{K}$ combination number is large). However, even though I used the newly-implemented functions and algorithms, the runtime only shortened by 25% (from 1m to 45s). As a result, I tried “Maximizer with Distance” method, which has similar uniqueness property as “Least Frequent” but only need one iteration of all k-node graphlet in the window. Expectedly, it optimizes the runtime even further (from 45s to 28s).

Besides, in order to make network databases, we have to get an initial knowledge of how many window samples are needed to cover a certain number of network nodes. Thus, in the test, I kept

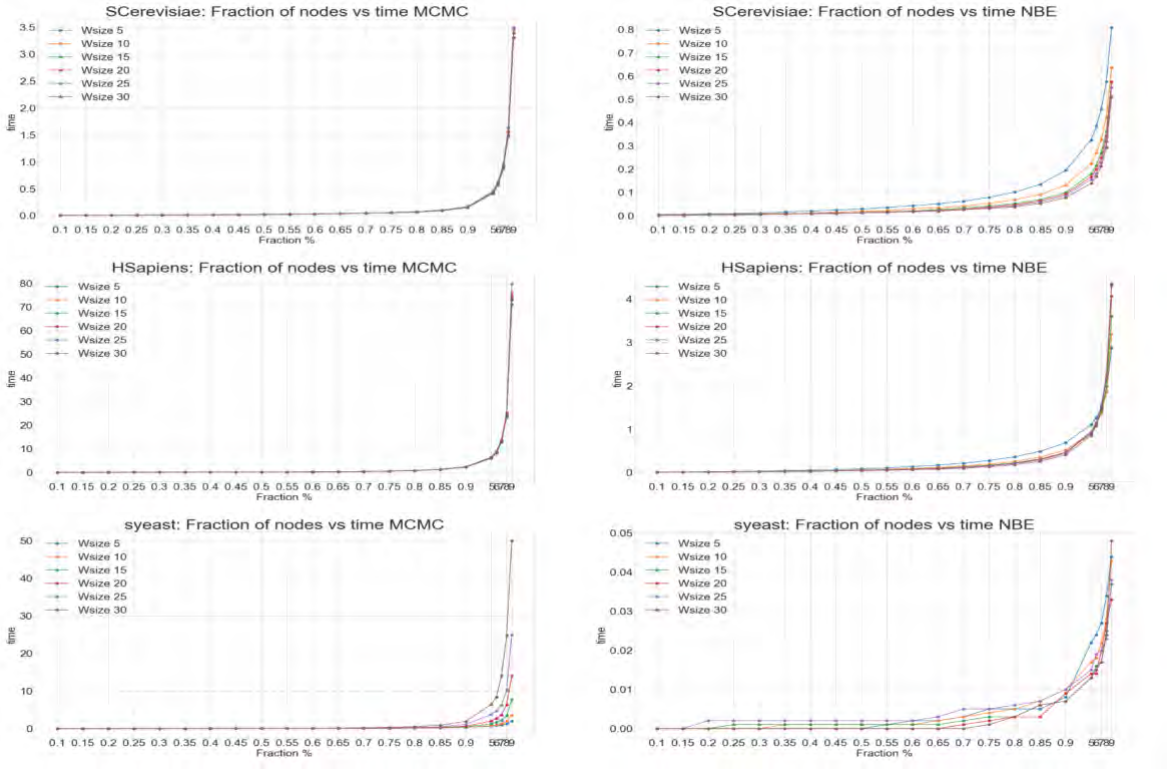
track of the needed sampling sizes and time for every 5% of the network nodes being covered. Followings are the results from three small networks (SCerevisiae, HSapiens, syeast), and three fairly large IID networks (IIDhuman, IIDmouse, IIDyest):



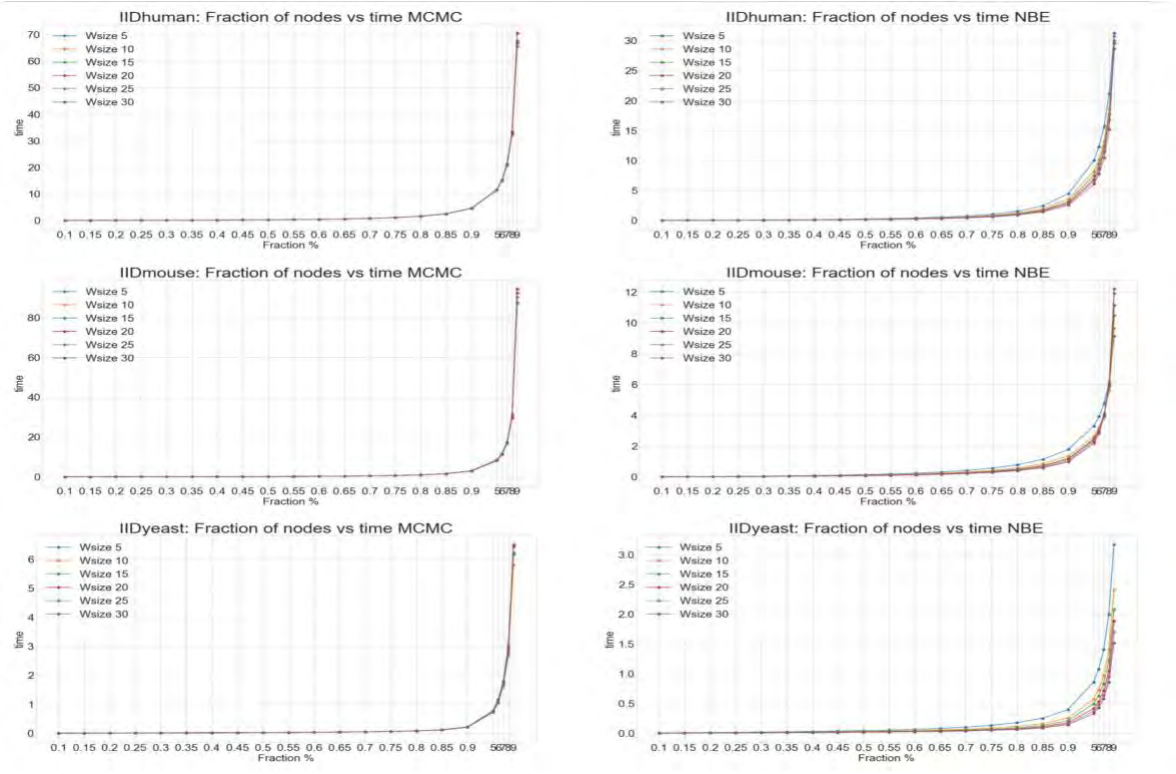
Plot 1: Required sample sizes for small networks



Plot 2: Required sample sizes for large networks



Plot 3: Sampling time for small networks



Plot 4: Sampling time for large networks

According to Plot 1&2, we can notice that using MCMC will result in a much larger required sample sizes compared to NBE. Besides, there is a much more noticeable difference between required number of windows for different window sizes by using NBE than MCMC. Based on Plot 3&4, we can observe the similar results that it usually takes longer for MCMC to sample out certain number of network nodes than NBE. Since the sampling time was obtained only through BLANT without using windowRep searching algorithm, we can get the statistics in a short time and use it to estimate how long it will take to find the windowReps by covering a certain number of network nodes.

Approaching to the end of this quarter, we focused on the stability property of windowReps, trying to use the sampled windowReps as the seeds for network alignments. Therefore, "Minimizer" method, which produces the best stable results, was chosen. In the current stage, we are interested to know what is the average lifecycle for each windowReps.

To summarize, during the past quarter, I have:

1. Implemented windowRep function in BLANT.
2. Added dynamic array into libwayne.
3. Found and implemented DFS like algorithm to make windowRep finding algorithm viable and faster if the combination number is large.
4. Overviewed and got the statistics for the required window sampling time and sizes to cover a certain number of network nodes.