Layout Algorithm

Rishi Rao

November 2021

1 Introduction

The algorithm used for this layout is calculated by defining a fitness function that is then optimized using python. The fitness function takes a test position for a node and compares it to all other nodes in the distribution, giving a value for the fitness level of that particular node at that position. The fitness function is defined by

$$\chi_i^2 = \sum_{j=0}^{\#ofnodes} (w_i * w_j)^{-a/b}(1 - \frac{2d_{ij}Fd_{ij}}{\sigma})^{a/b} \tag{1}$$

where F_{ij} is the fingerprint distance between i and j given by a fingerprint distance matrix, d_{ij} is the euclidean distance, w_i is the weight of matrix i, and a, b, σ are all adjustable parameters. The default for the test calculation was $a = 1$, $b = 2$ and $\sigma = 1$.

The weights for each node were calculated by summing over inverse fingerprint distance of the node to every other node in a neighbor list which contains every single neighbor for each node. This means that the neighbor list contains many repeats of the same node for nodes which are highly connected. Since the weights sum over these highly connected nodes more often, highly connected nodes will attract points that are close to them in fingerprint space towards the center. The equation used to calculate the weights are

$$w_i = \sum_{j=0}^{\#ofnodes} \sum_{k=0}^{\#ofneighbors(j)} \frac{1}{Fd_{ij}} \tag{2}$$

These two equations are minimized using python's minimize function provided with the scipy package. Each node is iterated through and optimized, with the entire graph being re-optimized 5 times before stopping.