1 The Node Similarity Measure

We use two different node similarity measures: the graphlet degree vectors similarity and sequence similarity.

1.1 Graphlet Degree Vectors and Topological Similarities

To determine topological similarity between two nodes in different networks, we use the similarity measure of nodes’ local neighborhoods, as described by Milenković and Pržulj. This measure generalizes the degree of a node, which counts the number of edges that the node touches, into the vector of graphlet degrees, that counts the number of graphlets that the node touches, for all 2-5-node graphlets. A graphlet is a small connected induced subgraph of a large network (e.g., an edge, a 3-node path, a triangle, etc.). Since it is topologically relevant to distinguish between, for example, nodes touching a 3-node path at an end or at the middle, the notion of automorphism orbits (or just orbits, for brevity) is used. By taking into account the “symmetries” between nodes of a graphlet, there are 73 different orbits across all 2- to 5-node graphlets. The full vector of 73 coordinates is the graphlet degree vector of a node that captures the node’s interconnectivities out to a distance of 4 (see for details).

Topological similarity between two nodes in different networks is computed as follows. For a node \( u \), \( u_i \) denotes the \( i \)th coordinate of its graphlet degree vector, i.e., \( u_i \) is the number of times node \( u \) touches an orbit \( i \). The distance \( D_i(u,v) \) between the \( i \)th orbits of nodes \( u \) and \( v \) is defined as:

\[
D_i(u,v) = w_i \left( \frac{\log(w_i) - \log(w_i+1)}{\log\left(\log(w_i+1)/\log(w_i+2)\right)} \right).
\]

where \( w_i \) is a weight of orbit \( i \) signifying its “importance” (see for details). The total distance \( D(u,v) \) between nodes \( u \) and \( v \) is defined as:

\[
D(u,v) = \sum_{i=1}^{72} D_i(u,v).
\]

The distance \( D(u,v) \) is in \([0, 1]\), where distance 0 means that graphlet degree vectors of nodes \( u \) and \( v \) are identical. Finally, the topological similarity, \( S(u,v) \), between nodes \( u \) and \( v \) is:

\[
S(u,v) = 1 - D(u,v) \quad (\text{see for details}).
\]

Clearly, a higher graphlet degree vector similarity between two nodes corresponds to a higher topological similarity between their extended neighborhoods.

1.2 Sequence Similarity

Sequence alignment is a way of arranging protein sequences to identify regions of similarity between them. Global sequence alignment algorithms attempt to align every amino acid in two sequences and are commonly used for similar sequences of proximately the same size. On the other hand, local sequence alignment algorithms attempt to find regions of local similarity between sequences and are generally useful for less similar sequences. BLAST (The Basic Local Alignment Search Tool) is a local sequence alignment algorithm that compares nucleotide or protein sequences and finds statistically significant regions of local similarity between these sequences.

For each aligned pair of sequences, BLAST returns an alignment score and a parameter called “expected value” (E-value). E-value describes the statistical significance of the alignment of two sequences, i.e., the probability that there exist another pair of sequences that have the same or higher alignment score than the two aligned sequences simply by chance. The lower the E-value (the closer it is to zero), the more significant the alignment is.

To calculate similarity between proteins in one network and proteins in the other network, we perform all-to-all BLAST alignment of these protein pairs and consider proteins with lower E-values to be more similar than proteins with higher E-values. That is, we define the sequence similarity \( S(v,u) \) between proteins \( v \) and \( u \) as:

\[
S(v,u) = 1 - E\text{-value}(v,u) \quad \text{if} \quad E\text{-value}(u,v) < 1 \quad \text{and} \quad S(v,u) = 0 \quad \text{otherwise}.
\]
2 Measures of the Alignment Quality

2.1 Edge Correctness

Recall that network alignment is the problem of finding the best way to “fit” $G$ into $H$ even if $G$ does not exist as an exact subgraph of $H$. However, it is not obvious how to measure the “goodness” of an inexact fit. One measure could be to assess the number of aligned edges—that is, the percentage of edges in $G$ that are aligned to edges in $H$. We call this the “edge correctness” (EC).

2.2 Statistical Significance of the Alignment.

Given alignment of two networks, $G(U, E)$ and $H(V, F)$, with the edge correctness (EC) of $x\%$, the probability $P_{al}$ of successfully aligning $k = [m_1 \times EC] = [m_1 \times x]$ or more edges by chance is the tail of the hypergeometric distribution:

$$P_{al} = 1 - \sum_{i=k}^{m_1} \left( \begin{array}{c} m_2 \\ i \end{array} \right) \left( \begin{array}{c} p-m_2 \\ m_2-i \end{array} \right),$$ (1)

where $n_1 = |U|$, $n_2 = |V|$, $m_1 = |E|$, $m_2 = |F|$, and $p = \frac{n_2(n_2-1)}{2}$ (i.e., $p$ is the number of pairs of nodes in $H$).

2.3 Statistical significance of shared GO terms.

To compute the statistical significance of the number of protein pairs sharing a Gene Ontology (GO) term, we use the standard model of sampling without replacement. Let $p$ be the number of all possible node pairs from $G$ and $H$ in which both proteins are annotated with at least one GO term, $m_2$ the number of pairs out of $p$ pairs in which both proteins share at least one common GO term, $m_1$ the number of pairs in the alignment in which both proteins are annotated with at least one GO term, and $k$ the number of pairs out of $m_1$ pairs in which both proteins share at least one common GO term. Then, the probability $P_{GO}$ of obtaining $k$ or more pairs of proteins that share at least one common GO term by chance is given with:

$$P_{GO} = 1 - \sum_{i=k}^{m_2} \left( \begin{array}{c} m_2 \\ i \end{array} \right) \left( \begin{array}{c} p-m_2 \\ m_2-i \end{array} \right).$$ (2)

3 C–GRAAL Algorithm (Pseudocode)

Algorithm 1 Align($G, H$), where $G=(V,E)$ and $H=(U,F)$; $v_i, v_j \in V$; $u_k, u_l \in U$; and $A[V] = U$ is alignment of nodes from $G$ to nodes from $H$.

Read in the node similarity matrix $S$;
Initialize alignment: $A \leftarrow \emptyset$;
Set $F_{al}$ to 0;
while $\exists$ an unaligned node $v_i$ in $G$ that is not in $A$
do
Step 1: $G, H$;
Step 2: $G, H$;
Step 3: $G, H$;
end while
Return alignment $A$;

Algorithm 2 Step 1($G, H$)

if $\exists$ a pair of nodes $(v_i, A[v_j])$ in $A$ that have at least one unaligned neighbor then
FAL $\leftarrow$ 1;
else
FAL $\leftarrow$ 0;
end if
Calculate the combined density of $v_i$’s and $A[v_j]$’s neighborhoods $cnd(v_i, A[v_j])$;
Select node $v_i$ with the maximum value of $cnd(v_i, A[v_j])$;
Based on $S$, align all unaligned neighbors of $v_i$ to all unaligned neighbors of $A[v_i]$;

Algorithm 3 Step2($G, H$),

if \(\exists\) two pairs of nodes \((v_i, A[v_i])\) and \((v_j, A[v_j])\) in \(A\) that have more than one common neighbor then

for all pairs of already aligned nodes \((v_i, v_j)\) do
  Calculate the number of their common neighbors, \(|CN(v_i, v_j)|\);
  Calculate the number common neighbors of nodes to which they are aligned, \(|CN(A[v_i], A[v_j])|\);
end for

Select a pair of nodes with the maximum value of \(|CN(v_i, v_j)|\), such that \(|CN(A[v_i], A[v_j])| > 0\);
Based on \(S\), align all unaligned common neighbors of \((v_i, v_j)\) to all unaligned common neighbors of nodes \((A[v_i], A[v_j])\);
else
  Step1($G, H$);
end if

Algorithm 4 Step3($G, H$)

for all unaligned nodes \(v_i\) do
  Based on \(S\), find unaligned node \(u_k\) that has the highest similarity to node \(v_i\);
  Align \(v_i\) to \(u_k\): \(A[v_i] = u_k\);
end for

References