UC Irvine

UC Irvine Previously Published Works

Title

BioFabric Visualization of Network Alignments

Permalink

<https://escholarship.org/uc/item/1h356808>

Authors

Desai, Rishi
Longabaugh, William JR
Hayes, Wayne

Publication Date

2019

DOI

10.1101/2019.12.18.881664

Peer reviewed

BioFabric Visualization of Network Alignments

Rishi M. Desai¹, William J.R. Longabaugh^{2*†} and Wayne B. Hayes^{1†}

*Correspondence:

wlongabaugh@systemsbiology.org

²Institute for Systems Biology,

Seattle WA, 98109-5234 USA

Full list of author information is available at the end of the article

[†](Address correspondence on visualization to WJRL; correspondence on alignment to WBH)

Abstract

Background: Dozens of global network alignment algorithms have been developed over the past fifteen years. Effective network visualization tools are lacking and would enhance our ability to gain an intuitive understanding of the strengths and weaknesses of these algorithms.

Results: We have created a plugin to the existing network visualization tool BioFabric, called *VISNAB: Visualization of Network Alignments using BioFabric*. We leverage BioFabric's unique approach to layout (nodes are horizontal lines connected by vertical lines representing edges) to improve understanding of network alignment performance. Our visualization tool allows the user to clearly spot deficiencies in alignments that cannot be detected through simply evaluating and comparing standard numerical topological measures such as the Edge Coverage (EC) or Symmetric Substructure Score (S^3). Furthermore, we provide new automatic layouts that allow researchers to identify problem areas in an alignment. Finally, our new definitions of *node groups* and *link groups* that arise from our visualization technique allows us to also introduce novel numeric measures for assessing alignment quality.

Conclusions: Our new approach to visualize network alignments will allow researchers to gain a new, and better, understanding of the strengths and shortcomings of the many available network alignment algorithms.

Keywords: network; visualization; alignment

Background

Network alignment is the process of finding a mapping between the nodes of two networks. Alignments have been used in social networks to identify social structure and de-anonymize supposedly anonymous networks [1], brain connectomes to aid the creation of a standardized brain atlas [2, 3], and biomolecular networks to identify functional relationships and to transfer information between species [4]. The increasing availability of protein-protein interaction (PPI) networks over the last two decades has spurred the development of dozens of new network alignment algorithms. Particularly, the alignments of PPI networks reveal key insights in protein function and similarity [4], which in turn offers better understanding of mechanisms of human disease [5] and the process of aging in humans [6]; this may enable the transfer of biological information across species.

The BioFabric network visualization tool was created to help visualize and analyze large and complex networks [7]. While the traditional way of drawing nodes as dots and edges as lines (a method we refer to as *node link diagrams*), BioFabric depicts nodes as *horizontal lines*, while edges are vertical lines; small squares at the top and bottom of an edge denote the nodes (horizontal lines) the edge connects. There is one node line per row, and one edge line per column, arranged on a strictly regular grid. Thus, edges *never* overlap, completely eliminating the inevitable “hairball” that results from depicting dense networks with a traditional node-link diagram. Also,

since links can originate and terminate anywhere along the appropriate node line segments, there is complete freedom to decide where a link is drawn. This flexibility for edge placement can be used to create many types of network visualizations that present meaningful semantic groupings of edges. The approach BioFabric uses has some similarities with “visibility representations” [8, 9], and a small example of using “nodes as lines” appeared in [10]. But BioFabric does not constrain nodes to be represented as discrete blocks and does not attempt to minimize link crossings at all. Indeed, in BioFabric it is possible to create useful network visualizations even when there are millions of intersections between node and edge lines.

There have been previous tools that have tackled the problem of visualizing network alignments [11, 12]. These tools still rely on variations of node-link diagrams and consequently suffer from the fundamental problems that plague this technique: alignments become harder to visualize as they become larger and more complex. The advantages provided by our new BioFabric plugin permit an entirely new approach to visualizing network alignments.

Our Contributions

BioFabric’s ability to group and order *both* nodes and edges in meaningful ways provides a new way to visualize and understand network alignments. Here we will introduce a new system of node and edge groupings, with a new associated layout, that provides unique insights into alignments. We also provide a new layout technique that leverages the linear ordering generated by node misalignments that can highlight problems in the alignment. Finally, we introduce some new metrics, inspired by our visualization approach, that can be used to assess the quality of an alignment.

Methodology

Overview and Nomenclature

Figure 1 illustrates how BioFabric depicts network alignments and introduces the color-based nomenclature we use in this paper. Figure 1A depicts the alignment using a traditional node-link representation. Nodes from the smaller *blue* network (top left) are aligned one-to-one onto nodes from the larger *red* network (top center), creating a combined network (top right) that is the union of both under one of many possible alignments. When network elements (nodes or edges) are matched in this procedure, we refer to them as *purple*. We refer to unmatched elements from the smaller network as *blue*, and unmatched elements from the larger network are called *red*.

Of course, in this example we are making the simplifying (and very common) assumption that *all* of the nodes in the smaller network are aligned to nodes in the larger network. In fact, in the most general alignment problem, we can also end up with *blue* nodes in the final alignment. In fact, VISNAB is fully capable of dealing with blue nodes in the final alignment. However, in our discussion of the technique, and in all the case studies we are discussing in this paper, we will deal with alignments with no blue nodes in the interest of simplicity. See the Appendix for additional information on how VISNAB handles blue nodes.

Figure 1B shows how BioFabric represents the combined network: the nodes are depicted as horizontal lines, and the edges are depicted as vertical lines. For this

example, to make the comparison more concrete, this figure departs from the actual BioFabric presentation in two ways. First, the nodes here are drawn purple and red to match their counterparts in the node-link diagram; second, the different classes of edges are distinctly grouped. In BioFabric, in order to allow the network alignment to scale, node colors are cycled in a strict pattern, and edges are organized on a uniform regular grid.

Node and Link Groupings

Consider the *rows* in Figure 1B: aligned purple nodes ($1::a$ through $4::d$) are grouped together exclusively in the top rows, while the unaligned red nodes (e and f) are grouped together exclusively in the bottom rows. Similarly, edges can be grouped in columns. As Figure 1C shows, there are *five* distinct classes of edges in the aligned network. Moving from left to right, we have purple *matched* edges, blue *orphan* edges, and three distinct classes of red *untouched* edges defined based upon the color of the two nodes incident on the edge: both purple, both red, or one of each. We refer to these five distinct classes as the *link groups* for a network alignment; they are enumerated in Table 1, along with the symbol we use to describe each one (e.g. **pRr**). For an alignment without blue nodes, these five link groups are a partition P of the set of edges in the union of the blue and red networks (where empty sets are allowed in P). The full enumeration of the seven link groups for the blue node case is provided in the Appendix.

BioFabric's link group separation allows visual quantification of the “common topology” discovered by an alignment. For example, one could argue that a better

Table 1 The enumeration of VISNAB's link groups for the case of an alignment with no blue nodes. The link group numbering corresponds to the numbering used for the full blue node case, given in the Appendix.

Link Group	Edge Color	Endpoint 1	Endpoint 2	Symbol
1	Purple	Purple	Purple	P
2	Blue	Purple	Purple	pBp
5	Red	Purple	Purple	pRp
6	Red	Purple	Red	pRr
7	Red	Red	Red	rRr

alignment would result by rotating the blue square 90° in either direction, combining the red and blue edges in the square's interior to a single purple edge; the BioFabric layout of the resulting alignment would immediately and visually depict the elimination of the red and blue interior edges.

The two major classes of purple and red nodes can be further subdivided into *node groups* based upon the types of edges incident upon the node; these twenty groups and their symbols are enumerated in Table 2 for the case with no blue nodes. For example, red nodes that are not singletons (groups 37-39) can be classified as having node neighbors that are: (i) only purple (all incident edges are **pRr**), (ii) purple and red (incident edges are **pRr** and **rRr**), or (iii) only red (all incident edges are **rRr**). As with the link groups, for an alignment without blue nodes, these 20 groups are a partition P of the set of nodes in the union of the blue and red networks (where empty sets are allowed in P). The full enumeration of the 40 node groups for the blue node case are also provided in the Appendix.

Network Merge

Most attempts to visualize an alignment provide side-by-side node-link diagrams of the input networks, with edges drawn between them to depict the alignment. The result rarely provides intuition about common topology discovered by the alignment. Instead, we *merge* the input networks (cf. Figure 1A, far right), characterizing the result with our node and link groups. Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two

Table 2 The enumeration of VISNAB's node groups for the case of an alignment with no blue nodes. The node group numbering corresponds to the numbering used for the full blue node case, given in the Appendix.

Node Group	Node Color	Incident Edges	Symbol
1	Purple	None	(P:0)
2	Purple	Purple	(P:P)
3	Purple	pBp	(P:pBp)
6	Purple	pRp	(P:pRp)
7	Purple	Purple, pBp	(P:P/pBp)
10	Purple	Purple, pRp	(P:P/pRp)
11	Purple	pBp, pRp	(P:pBp/pRp)
14	Purple	Purple, pBp, pRp	(P:P/pBp/pRp)
17	Purple	pRr	(P:pRr)
18	Purple	Purple, pRr	(P:P/pRr)
19	Purple	pBp, pRr	(P:pBp/pRr)
22	Purple	pRp, pRr	(P:pRp/pRr)
23	Purple	Purple, pBp, pRr	(P:P/pBp/pRr)
26	Purple	Purple, pRp, pRr	(P:P/pRp/pRr)
27	Purple	pBp, pRp, pRr	(P:pBp/pRp/pRr)
30	Purple	Purple, pBp, pRp, pRr	(P:P/pBp/pRp/pRr)
37	Red	pRr	(R:pRr)
38	Red	rRr	(R:rRr)
39	Red	pRr, rRr	(R:pRr/rRr)
40	Red	None	(R:0)

networks with $|V_1| \leq |V_2|$. We define a pairwise global alignment from G_1 to G_2 is an injective function $a : V_a \rightarrow V_2$; every node in $V_a \subseteq V_1$ is mapped to a distinct node in V_2 , and V_a is allowed to be a strict subset of V_1 , indicating some nodes in V_1 (the blue nodes) may not be mapped. The merged network $G_{12} = (V_{12}, E_{12})$ consists of the nodes and edges in the two networks. All aligned nodes $u \in V_a$, $v \in V_2$ with $a(u) = v$ are combined into one node n and added to V_{12} . A combined node n is labeled in the format $u::v$. The unaligned nodes in V_1 and V_2 are also added to V_{12} . E_{12} consists of all the edges in E_1 and E_2 . An *aligned* edge is an edge $(u_1, u_2) \in E_1 : (a(u_1), a(u_2)) \in E_2$. Aligned edges (link group **P**) are only represented *once* in E_{12} . Hence, the total number of edges $|E_{12}|$ varies with different alignments.

New Metrics

Most topology-only network alignment approaches currently in use perform rather poorly across a wide range of biological test sets, particularly when the alignment is evaluated using **Node Correctness** (NC). Traditionally NC is the fraction of nodes $u \in V_1$ aligned correctly when the correct alignment is known; we extended NC for alignments with unaligned nodes $u \in V_1$. Let $a'(u)$ be $a(u)$ if $a(u)$ is defined and u otherwise. Given an alignment $a : V_a \rightarrow V_2$ and the known correct alignment $a_c : V_c \rightarrow V_2$ with $V_a, V_c \subseteq V_1$, our NC measure is defined as the fraction of nodes $u \in V_1 : a'(u) = a'_c(u)$.

In fact, slight differences in NC values, even if statistically significant, are arguably not very informative to assess pure topologically-driven alignments. We demonstrate here that other ways of assessing alignment performance can provide richer insights into the behavior of different techniques.

For example, if two nodes u and v in any graph G have an identical neighbor set (excluding the edge (u, v) if it exists), then it is impossible to distinguish between them using topology alone; thus, an alignment should not be penalized for swapping u and v if one of the two potential swaps is the “correct” one. Similarly, if k nodes all share the same neighbor set (excluding all edges between the k nodes themselves), any permutation of these k nodes should be scored equally if one of the permutations yields the “correct” alignment. This observation leads us to adapt a new **Jaccard Similarity** (JS) measure in the context of network alignment, as follows:

For some network $G = (V, E)$, define the neighborhood of node z_1 in G to be $N_G(z_1) = \{z_2 \in V : (z_1, z_2) \in E\}$. For nodes $x, y \in V$, let $N_G(x, y)$ be the neighborhood of x disregarding y , and let i_{xy} be a corrective term accounting for a possible edge between the two, as follows: if $y \in N_G(x)$ then $N_G(x, y) = N_G(x) - y$ and $i_{xy} = 1$, else $N_G(x, y) = N_G(x)$ and $i_{xy} = 0$. Our extended JS definition $\sigma_G : V \times V \rightarrow [0, 1]$ between two nodes is defined as:

$$\sigma_G(x, y) = \frac{|N_G(x, y) \cap N_G(y, x)| + i_{xy}}{|N_G(x, y) \cup N_G(y, x)| + i_{xy}}$$

Intuitively, two aligned nodes score a 1.0 if they share an identical set of neighbors (ignoring a possible self-link between them) and are therefore impossible to distinguish using topology alone, and so a topology-driven alignment score shouldn't

penalize the misalignment of such pairs. (Note that when x and y are both singletons, we define $\sigma_G(x, y) \equiv 1.0$ to avoid dividing by zero.)

Given node sets $V_a, V_c \subseteq V_1$, an alignment $a : V_a \rightarrow V_2$, and the correct alignment $a_c : V_c \rightarrow V_2$, our JS measure for the given alignment a , with respect to the correct alignment a_c , in the case where $|V_a| = |V_c| = |V_1|$ is defined as:

$$JS(a) = \frac{1}{|V_1|} \sum_{u \in V_1} \sigma_{G_2}(a(u), a_c(u))$$

The JS measure for the “blue node” case, where $|V_a|, |V_c| \leq |V_1|$, is detailed in the Appendix. Significantly, Jaccard Similarity does *not* penalize a misalignment if the two nodes are topologically indistinguishable. Note also how JS gives “partial credit” for alignments where the nodes have nearly the same sets of neighbors.

The importance of node and link groups to VISNAB visualizations inspired us to design two new metrics that can be used to assess the performance of an alignment in the presence of a known correct alignment ($NC = 1$) between the two networks. For this, we use the node group and link group equivalence relations over the groupings of nodes and edges in an aligned network that we have developed. We can do this by comparing how the distribution of nodes and edges in the given alignment across the groupings compares to the distribution found in the correct alignment. To this end, we have created the **Node Group Similarity** (NGS) and **Link Group Similarity** (LGS) metrics. Both NGS and LGS are calculated using a vector representing the proportion of nodes and links distributed among the twenty node groups and five link groups, respectively. For example, the NGS vector $\mathbf{r}$ for k node groups $ng_1, ng_2, \dots, ng_k$ where $|V_{12}| = |V_1| + |V_2| - |V_a|$ is

$$\mathbf{r} = \left\langle \frac{|ng_1|}{|V_{12}|}, \frac{|ng_2|}{|V_{12}|}, \dots, \frac{|ng_k|}{|V_{12}|} \right\rangle$$

The angular similarity between the vector of the given alignment and the vector of the known correct alignment is calculated. For vectors with only positive elements, angular similarity $= 1 - 2\theta/\pi$, where θ is the angle between the vector of the alignment and the vector of the correct alignment. Angular similarity was chosen (instead of e.g. cosine similarity) because it exhibits more rapid decay from the optimum value in the case of small angles.

All of these metrics, as well as other common measures, are displayed by VISNAB in a dialog when the user provides the correct alignment file as an optional part of the setup steps.

Node and Link Group Layout Algorithm

Once we have created a merged network, we must provide an informative layout of the network that organizes the nodes and edges into a grouping framework that aids understanding the alignment. All BioFabric network layouts are fully defined by creating a linear ordering of the nodes, and a linear ordering of the edges. The default technique for laying out nodes is simple: within each node group it creates a linear ordering of the nodes using a breadth-first search from the node of highest degree, where neighboring nodes are visited in the order of highest to lowest degree.

This basic technique, which uses a single queue for the breadth-first search, can be modified to create a layout that organizes the nodes using the alignment-based groupings we have defined. Specifically, we adapted it to use a *separate* queue for each of the twenty node groups. The resulting layout retains much of the same overall structure as the default algorithm, while ensuring that nodes are well-organized in each node group band. As nodes are visited during the search, they are added to the queue corresponding to their node group, and these node group queues are processed in the order listed in Table 2. A more complete description of the algorithm is provided in the Appendix.

Finally, if the correct alignment is provided, the user can choose to have correctly and incorrectly aligned nodes laid out separately in different node groups. The user can choose the criterion for the correct alignment to be based either on the traditional NC measure, or on our JS measure. If JS is chosen, the user can set the threshold value $\beta \in [0, 1]$, so, in the case where $|V_a| = |V_c| = |V_1|$, an aligned node $u::v$ is denoted correct if $\sigma_{G_2}(v, a_c(u)) \geq \beta$. If JS is chosen, the condition for correctly aligned nodes for the “blue node” case, where $|V_a|, |V_c| \leq |V_1|$, is detailed in the Appendix.

The default edge layout algorithm, using a slightly modified version of the existing link group feature described in [7], can be used to organize the edges into the five link groups. The new modification allows edges that are tagged with a particular relation to be grouped contiguously on a *per-network* basis; previously, this could only be done on a *per-node* basis. Using this new modification, one of five tags signifying the edge’s link group (Table 1) is assigned to each edge in the network. The edges are then partitioned based on these tags into five link groups, like those in Figure 1. Consequently, the upper-left corner of the layout contains the purple nodes that are either singletons or have only purple edges (**P:P**).

Alignment Cycle Layout for Self-Alignments

A common approach for assessing the performance of alignment algorithms is to align a subset of a network to itself, where the larger network contains the same number of nodes but a larger number of edges [4, 13]. To allow researchers to better understand these analyses, we developed a special network alignment layout method that highlights alignment problems. Our technique leverages the fact that if a network is aligned to itself, then it can be viewed as a set of *cycles* each covering the set of *nodes*. Correctly aligned nodes will have a cycle length of one, $u \rightarrow u$, but a misalignment of $u \rightarrow v$, and $v \rightarrow u$, will create a cycle of length two. More severe misalignments create longer cycles. Figure 2 shows a small example of this approach.

Since a node layout in BioFabric is just a linear ordering of nodes, these alignment cycles provide a natural basis for specifying this ordering; the resulting layout vividly portrays the nature of the alignment. We start by placing a node in the first row, then use the alignment cycle containing that node to specify the row ordering of all remaining nodes in the cycle. The decision of how to order these cycles is again derived from a modification of BioFabric’s breadth-first search default layout algorithm [7]. For the alignment cycle layout, *all* the nodes in the cycle for a neighbor are simply placed first before the next neighbor is processed.

Though this technique is easiest to picture when a network is aligned to itself, the method has been generalized to handle network alignments from one network to another with *more* nodes. In this case, the alignment creates a set of *paths* instead of cycles, since not all nodes in the large network will map to a node in the small one. With paths, the algorithm places all nodes belonging to a path when any node in the path is first encountered in the search. Finally, the user can provide a mapping if the nodes in the two networks are in different namespaces (e.g. one network uses *Saccharomyces Genome Database* (SGD) IDs for node names, while the other network uses *ENTREZ* IDs).

Interface with BioFabric

BioFabric Version 2 (now in beta release) provides a new plug-in architecture that allows developers to add new features to the program. A plug-in is written in Java, and when the .jar file containing the compiled code is placed in a directory specified by the user, the new functionality is available in BioFabric's **Tools** menu. Our VISNAB plug-in uses this new architecture to provide the new functions that allow the user to load in two networks, an alignment file, and an optional correct reference alignment, and then process and lay out the desired view of the alignment. The plug-in also uses the new node and link annotation feature that is now being introduced

in BioFabric Version 2. This feature allows the user to specify spans of nodes or links that are then highlighted by drawing colored rectangles in the background.

Results

Case Study I: Simple Network Comparison

VISNAB can be used to directly compare two networks when aligning a network to itself. An example of this is where the nodes are proteins from the same organism and the networks are from two different studies. Our new node and link group definitions can provide visual insights into how the two networks compare. Notably, this technique can also be used to compare *any* two networks, as long as a 1:1 mapping of nodes between the two networks is provided.

Figure 3 shows a “correct” alignment between two datasets describing the *S. cerevisiae* (baker’s yeast) PPI network. The smaller *Yeast 2K* is derived from a network generated from data in [13] and used in [4], and contains 2,390 nodes and 16,127 edges, while the larger *SC* (BioGRID (v3.2.101, June 2013) [14]) contains 5,831 nodes and 77,149 edges. The networks are “aligned” based upon simply matching node names. Note these networks used different protein naming conventions; the Appendix provides details on how we created this correct alignment.

The top of Figure 3 shows the entire merged network, while the detail below it displays the subnetwork with purple and blue edges. The lowest detail shows the distribution of nodes into our node groups, which are denoted by the horizontal colored bands. This display uses a recently added BioFabric Version 2 feature: node and link annotation displays.

Note how the five link groups and twenty node groups quickly reveal how these two networks compare. We immediately see that the number of nodes in the larger

network are more than double the number in the original, as the purple nodes are in the colored bands down through the **(P:P/pBp/pRp/pRr)** (peach colored) node group; the node below that are red nodes only in the larger network. The edges in the two leftmost shaded link group bands (**P** and **pBp**) represent *exactly* the links in the *Yeast 2K* network. If the larger network were to be treated as a later, more complete, and more accurate survey, then the edges in group **pBp** represent the *false positives* from the earlier survey. The red edges in middle link group **pRp** are new edges from the more recent survey between old nodes from the original survey, while the edges in the right 40%, in the **pRr** and **rRr** link groups, represent new edges incident on at least one new red node.

Studying the relative sizes of the node groups also provides insights into how the networks differ. One thing to note right away is that the six node groups we do not expect to appear are in fact absent: **(P:0)**, **(P:pBp)**, **(P:pRp)**, **(P:pRr)**, **(P:pRp/pRr)**, and **(R:0)**. We expect this because these protein-protein interaction networks do not have any singleton nodes. Thus, for example, a node in the **(P:pBp)** class would require that the node be a singleton in the larger network. Conversely, a purple node with only red edges would require that the node be a singleton in the smaller network.

Another significant fact that is apparent by the node group distribution is that **over 90%** of the purple nodes are in the three node groups that have red edges going to both purple and red nodes: **(P:P/pRp/pRr)** (pink), **(P:pBp/pRp/pRr)** (powder blue), and **(P:P/pBp/pRp/pRr)** (peach). In contrast, there are only a tiny number of aligned nodes that have red edges going to *only* aligned or *only* unaligned nodes. Again, this seems consistent with a wider survey that would be expected to mostly find new interactions between both previously surveyed proteins as well as between those proteins and new proteins. The red nodes also show the same pattern of mostly ($\sim 80\%$) having nodes with interactions to both new and old proteins. Finally, note how group **(P:pBp/pRp/pRr)** is the smallest of the major purple node groups. This tells us that there were fewer proteins in the original network where all the old edges were considered false positives, again a result that makes sense.

Case Study II: Visualizing Common Topological Measures

There are several measures to assess the topological quality of an alignment. Depending on the context, these measures quantify how much topological similarity between the two PPI networks an alignment has exposed. Let $E_a = \{(u_1, u_2) \in E_1 : (a(u_1), a(u_2)) \in E_2\}$ denote the edges in G_1 aligned to edges in G_2 , i.e., purple edges. The **EC** measure (variously called Edge Coverage, Conservation, Correctness, or Correspondence by different authors) is the fraction of edges in the smaller network that are aligned to edges in the larger network: $EC(a) = |E_a|/|E_1|$. Let $\hat{E}_a = \{(v_1, v_2) \in E_2 : \exists u_1, u_2 \in V_1 \wedge a(u_1) = v_1 \wedge a(u_2) = v_2\}$ denote the set of edges of G_2 induced on its aligned nodes. **Induced Conserved Structure (ICS)** is the ratio of aligned edges to induced edges: $ICS(a) = |E_a|/|\hat{E}_a|$. However, **EC** and **ICS** have the shortcoming that they can be high if the alignment maps sparse regions of one network to dense regions of the other. In the extreme, if G_2 is a clique, any alignment has $EC = 1$. To overcome this drawback, [15] devised the **Symmetric**

Substructure Score (S^3), the ratio of aligned edges to all edges with aligned endpoint nodes:

$$S^3(a) = \frac{|E_a|}{|E_1| + |\hat{E}_a| - |E_a|} = \frac{\text{purple edges}}{\text{edges of all colors between purple nodes}}$$

Since BioFabric lays out node rows and link columns on an absolute regular grid, the *proportions* of the widths of various node and link groups allow the user to rapidly visualize these topological metrics at a glance. For example, S^3 can be estimated by comparing the *width* of the first link group (purple edges) over the *width* of the first three link groups (all links between purple nodes):

$$S^3 = \frac{|\mathbf{P}|}{|\mathbf{P}| + |\mathbf{pBp}| + |\mathbf{pRp}|}$$

Looking at the top view of Figure 3, we visually estimate that this alignment has an S^3 score of about 1/4 (actual value: .248). In the same fashion, EC is the width of the purple link group over the width of the purple and blue link groups:

$$EC = \frac{|\mathbf{P}|}{|\mathbf{P}| + |\mathbf{pBp}|}$$

We can visually estimate EC to be about 2/3 (actual value: .689). Finally, ICS is the width of the purple link group over width of the purple group and red group with purple endpoint nodes:

$$ICS = \frac{|\mathbf{P}|}{|\mathbf{P}| + |\mathbf{pRp}|}$$

We can visually estimate ICS to be around 3/10 (actual value: .280). Thus, BioFabric's node and link groups provide intuition on abstract topological measures.

Case Study III: Visualizing Performance of Objective Functions

Choosing the best objective function to apply to a particular network alignment is currently a black art, and providing the researcher with a visual tool to assist in understanding the properties and performance of various objective functions helps to gain insights into this difficult problem.

Indeed, BioFabric's presentation of network alignment allows the researcher to do just this. Figure 4 depicts four alignments between the *Yeast 2K* and *SC* networks previously introduced. The first row of the figure is the same gold-standard, correct alignment shown in Figure 3. For the other alignments, we used SANA [16], which allows us to select the objective functions we wish to use to create an alignment. These other three alignments were generated by running SANA for ten hours each (long enough for the random search algorithm to converge to a near-optimal score of the chosen objective function), optimizing the following objective functions: the second-row alignment was generated with an objective function utilizing only **Importance** (I) [17]; the third with .03 weight given to S^3 and .97 weight given to I ; and the fourth utilizing only S^3 .

In addition to viewing the BioFabric plots of the four alignments, we can also look at our new *NGS*, *LGS*, and *JS* metrics for them, as well as the traditional *NC*, S^3 , and Resnik semantic similarity scores [18, 19, 20] calculated by FastSemSim [21]. These scores are shown in Table 3. The highest S^3 value and the highest *NC* value are often the measures used to identify the “best” alignment. As the table shows, the pure S^3 version provides the highest of both of these scores. Yet this “best” *NC* score is only 2.1%, which is not a very inspiring result. Even our new *JS* score of 6.9%, which is tolerant of mismatches that are topologically similar, is still very low.

Simply *looking* at the four BioFabric plots in Figure 4 makes us question calling the pure S^3 version the “best” alignment of these two networks. Recalling that S^3 is the width of the first link group over the width of the first three link groups, we can instantly see that (i) the pure S^3 version has a much higher value than any other alignment, even the correct alignment, and (ii) it achieves this by creating an alignment that forces *far* more edges into the two rightmost link groups **pRr** and **rRr**. Particularly worrisome is the very large increase in the size of **rRr** compared to the correct alignment; this set represents the edges and nodes that have been completely omitted from the alignment.

This fact that high-degree nodes were relegated to the red untouched nodes provides a clue to a way to improve the situation. Perhaps using Importance (in which the highest degree nodes tend to align to each other) as an objective function can remedy the problem? This approach gives rise to the “Pure Importance” version in row two of Figure 4. Indeed, this version has link groups **pRr** and **rRr** on the right end that appear to be *much* closer in size to the correct alignment, and the center link group **pRp** is closer to the correct alignment as well, albeit larger. But the

Table 3 Scores for the four alignments between Yeast 2K and SC.

Alignment	NGS	LGS	NC	JS	S^3	Resnik
Correct	1.00	1.00	1.00	1.00	0.25	9.63
$1.0 * I$	0.61	0.79	0.00042	0.021	0.0043	3.16
$.03 * S^3 + .97 * I$	0.87	0.80	0.018	0.057	0.27	3.62
$1.0 * S^3$	0.64	0.46	0.021	0.069	0.55	3.61

leftmost group of purple **P** edges is so thin that it is almost non-existent. Hence, pure Importance yields an S^3 value of merely .0043.

So, perhaps a simple linear mixture of the two objective functions can provide a compromise between these two extremes? To investigate this, we ran a series of alignment runs that used a linear mixture of the two objective functions, and nine combinations were scored. The scores for all nine are included in the Appendix. The mixture $(.03 * S^3) + (.97 * I)$ is shown in the third row, and provides a reasonable *visual match* of the gold-standard alignment in terms of the distribution of edges between the five link groups. This visual similarity is also borne from the metrics shown in Table 3.

Scanning across a variety of mixture values, this particular mix produced an alignment with a high *NGS*, a reasonably high *LGS*, a non-zero *NC*, and a respectable *JS* value. It also has the closest S^3 value to the gold-standard, plus the highest functional similarity, i.e. the most biological relevance per the Resnik score, of all the alignments, though only marginally.

Another way to compare performance is by studying the node group distribution under the different objective functions, as shown in Figure 5. Notably, the “best” mixed alignment’s four largest node groups are the same as the four largest node groups of the correct alignment. The two largest purple node groups of each, **(P:P/pRp/pRr)** (pink) and **(P:P/pBp/pRp/pRr)** (peach), show a rich mixture of incident purple, red, and (in the latter group) blue edges. A significant difference between the two is the thicker bands above the pink **(P:P/pRp/pRr)** region for the mixed alignment compared to the correct alignment. The majority (76%) of nodes in those bands have no **pRr** incident edges, which reflect the fact that although there are *more* **pRr** edges in the mixed alignment (see Figure 3),

those edges are concentrated across a *smaller* fraction of the purple nodes. Specifically, 23% of purple nodes in the mixed alignment have no **pRr** incident edges, compared to just 5% in the correct case. Table 10 in the Appendix provides the exact sizes of the different node groups for the four alignments.

Case Study IV: Finding Protein Cluster Misalignments

While looking at metric values for a particular alignment can provide some general idea of how an alignment performs, there is value in being able to quickly spot major problems in an alignment where a gold-standard alignment is available, and then be able to understand what the problem is.

Figure 6 shows a yeast network, *Yeast0*, of 1,004 nodes and 8,323 edges aligned to a network with the same set of nodes, but with 20% more (9,987) edges, *Yeast20*. This network is part of the *noisy yeast* variations on the network in [13]. The edges in the smaller network are a strict subset of the larger set. This dataset has been used in several previous studies [15, 16]. The particular alignment we are using here was part of an ensemble generated by SANA [16], but was unusually poor compared to other alignments in the ensemble; this lead us to ask what was so different about this poor alignment? This visualization is created using the same network merge technique described in **Network Merge**, above. The nodes in this network are labeled to represent the alignment result, e.g. *a::a* for a correct alignment, and *a::b* for an incorrect alignment. Edges are labeled **P** (for purple edges), **pBp** (blue), or **pRp** (red).

To be able to spot alignment problems, we view the network using our new Alignment Cycle layout method, described above. This layout does not partition nodes into node groups, but still generates link groups. Since the set of nodes is the same for both networks, there are no red nodes, so only three of the five link groups are present. For this type of presentation, we employ BioFabric’s *per-node* link group layout to separate links into the three distinct groups within each node’s dedicated *node zone*. This is in contrast to the *per-network* approach we used previously, which creates single global regions for each link group.

The large uncolored stretches of Figure 6 indicate that most of the nodes—particularly high-degree ones—are correctly aligned. However, cycles of misalignment, even those of length two, are shown with alternating orange and green blocks, generated using the link annotation feature. In order to understand these misalignments better, we zoom into each region to study them more closely.

One interesting region includes a misalignment cycle containing 15 nodes, as shown in Figure 7A. A node-link diagram version created using Cytoscape [22]

is shown, where the protein-protein interactions are blue, and directed red edges represent the alignments. The BioFabric representation is shown at the top, with an inset detail shown as well. In addition to the large 15-node misalignment cycle on the left half of the node-link diagram, there are also correct alignments and small cycles of length two and three.

This is clearly a single protein cluster; in fact, it is some of the protein components of the mitochondrial ribosomal protein of the large subunit. (Note that all characterizations of proteins in this paper were obtained from the SGD [23].) While not fully connected, it does approximate a clique, so the misalignments found here are unsurprising, given the objective function uses topology alone.

Crucially, we can instantly interpret the context of the misalignment with the BioFabric version of the network: the repeating pattern of “edge wedges” is the telltale sign in BioFabric of a clique when the nodes in the clique are laid out contiguously. Remembering that the order of the node rows has been determined using the order of the alignment cycles, we see that in this case the node order still maintains the typical clique pattern; thus the misalignments are actually *contained within the clique*. Since nodes within a clique are topologically indistinguishable, their misalignment is unavoidable (unless more information such as gene sequence data is used in the alignment objective function), and should not be penalized for an alignment which is driven by topology alone.

A more profound alignment problem is shown in Figure 7B. This misalignment involves 32 nodes in three alignment cycles. As the red edges in the node-link diagram shows, the alignment has incorrectly aligned one *entire* protein complex

with a completely different complex! On the left are proteins of the CPF cleavage and polyadenylation factor, while on the right are the components of the RSC chromatin remodeling complex.

This problem stands out in the BioFabric version, where the user typically studies and compares the “edge wedge” shapes for nodes to better understand the network structure. While a typical clique pattern (as shown in Figure 7A) has edge wedges with a 45° lower margin, these wedges have a steeper, 60° lower margin; the edges for each node are actually incident on *alternating* nodes in this run. The explanation of this pattern is that the nodes belonging to the two separate protein clusters are intercalated in this run of nodes that have been laid out using the alignment cycles. Thus, our Alignment Cycle layout technique makes this problem stand out just by looking at it.

Even if one would argue that the traditional node-link diagram shows these effects well, the views shown had to be meticulously hand-crafted to illustrate the structure once we knew what we were looking at, while the BioFabric representations were automatically laid out. As we show in the Appendix, even a complex misalignment involving a cycle of *four* protein clusters can be quickly spotted.

Discussion

Given the difficulty of viewing the quality and characteristics of network alignments using traditional node-link diagrams, it is unsurprising that researchers have concentrated on evaluating alignments by comparing and optimizing certain numerical metrics. But to be able to gain some broad intuition about network alignments, having effective and well-organized visualization techniques can provide huge benefits in understanding the problems.

While still providing the deep intuition of the node-link approach (albeit with the nodes depicted as lines), BioFabric additionally give the user the ability to group and order *both* nodes and edges in meaningful ways. This gives researchers a new way to tackle the problem of understanding network alignments. The organized grouping of *edges* that is made possible by nodes-as-lines is particularly unique. As we have shown, simple visual estimates of proportions of classes of link groups provides intuition into abstract measures like S^3 . Furthermore, as our Alignment Cycle layout demonstrates, it is possible to create new, specialized layout algorithms that exploit the linear ordering of nodes in ways that make it possible to quickly spot alignment pathologies.

Because of these features, VISNAB is a viable tool for studying objective functions and their performance. As discussed in [16], we strongly contend the network alignment community should focus not on developing new network alignment algorithms, but rather on devising more effective objective functions. Just as we experimented with combining S^3 and Importance, researchers should design custom objective functions for their specific needs to yield optimum results. To accomplish this task, we propose VISNAB as a tool researchers should utilize to help design objective functions.

For example, as we have shown in Case Study III, researchers can work towards creating objective functions that distribute edges and links into appropriate proportions of our identified groupings. As was shown in Case Study I, the sizes of these

groupings can reflect the character of the alignment problem at hand, e.g. a pair of experiments where the larger network represented a deeper, more thorough investigation of the system, where new elements were added, and false positives were identified. So an objective function that worked to take this topology into account would be an improvement. Note that this approach can be useful *even in the absence of a gold standard alignment* for a given problem, as the overall proportions of the groupings could arguably apply to other alignments in the same class of problems. In other words, target vectors for the *NGS* and *LGS* scores could be estimated a priori for a class of problems.

Limitations

BioFabric’s presentation of “nodes as lines” is unfamiliar for network researchers who have grown accustomed to visualizing nodes as points and can take some time getting used to. For networks small enough, the traditional node-link diagram approach is adequate, so it would be beneficial to allow the user to select a subset of the BioFabric network and view it using the traditional presentation. Going the other way, taking a subset of a traditional node-link presentation and viewing it using BioFabric, would be valuable as well. But this capability is not currently available.

Another shortcoming of the current VISNAB implementation is that the ordering of the node groups is fixed, per the order in Table 2. A different order may be more helpful in visualizing a particular alignment, so being able to dynamically reorder the node groups would be a useful feature.

Finally, just like the existing *NC* score, our new *NGS*, *LGS*, and *JS* scores depend upon the availability of a gold-standard alignment. Thus, they can really only be used to evaluate the performance of alignment algorithms and objective functions when the correct answer is already known. To be incorporated into a useful objective function, it would be necessary that for certain classes of alignment problems, target vectors for the *NGS* and *LGS* scores could be estimated a priori. It is also an open question about the difficulties of developing objective functions based upon incremental evaluation for these measures.

Future Work

An emerging area of investigation is the multiple network alignment problem, e.g. [24]. In these types of alignments, the fraction of the full set of networks that share a particular edge or node in the alignment is a crucial piece of information. This suggests that creating node and link groups for grouping elements with similar overlap percentages would be helpful in visualizing the quality and characteristics of a multiple network alignment.

Our new tool provides a platform to implement other new visualizations for exploring network alignments. Constructing and visualizing subgraphs of the full merged network (from **Network Merge**, above) can provide valuable insights as well. For example, one additional view that we support is the Orphan Edge layout, which shows the subgraph of G_1 that consists all the nodes connected by the orphan blue **pBp** edges, plus the first degree neighbors of those nodes and corresponding connecting edges. In aligned networks where $E_1 \subseteq E_2$ (e.g. the *noisy yeast* network

used in Case Study IV), blue edges will not exist at all in a correctly aligned network. Thus, viewing the context of the orphan blue **pBp** edges in the network can provide insights into alignment problems.

Similarly, we can visualize the common subgraph $CS_a = (V_1, E_a)$ of the alignment by displaying only the aligned purple **P** edges and the purple nodes they are incident upon. If this subgraph's nodes and edges are laid out using BioFabric's default layout [7], the various connected components of CS_a can be assessed. One way some researchers evaluate alignments is to determine if the alignment has a common subgraph with large connected regions [4, 15]. To this end, the user can visually estimate the components of the measure **Largest Common Connected Subgraph** (LCCS) using the heights of the nodes and widths of the edges of the connected components.

Conclusions

VISNAB provides a novel, powerful new way to visualize network alignments, and will allow researchers to gain a new and better understanding of the strengths and shortcomings of the many available network alignment algorithms.

Appendix

Link and Node Groups With Blue Nodes

In the main text, for simplicity, we limited the discussion to the common case of aligning of network G_1 onto network G_2 when every node in G_1 is aligned onto a node in G_2 . In the nomenclature we have introduced, this is an alignment “without any blue nodes”. Table 1 and Table 2 enumerated the possible link and node groups for these alignments. However, VISNAB is capable of handling alignments where unaligned blue nodes are permitted. In that case, the five link groups expand to seven, adding in groups three and four, which account for the case where blue edges are incident on blue nodes. Table 4 enumerates all possible link groups in the presence of blue nodes in the alignment.

Table 4 Expansion of Link Groups When Unaligned (Blue) Nodes are Present.

Link Group	Edge Color	Endpoint 1	Endpoint 2	Symbol
1	Purple	Purple	Purple	P
2	Blue	Purple	Purple	pBp
3	Blue	Purple	Blue	pBb
4	Blue	Blue	Blue	bBb
5	Red	Purple	Purple	pRp
6	Red	Purple	Red	pRr
7	Red	Red	Red	rRr

In a similar fashion, the twenty node groups that can be present in an alignment without blue nodes expands to forty possible groups when blue nodes are allowed. Note how, for example, Node Group 3 (see Table 2) splits into three distinct groups (3, 4, and 5) with blue nodes, since blue edges can be incident on blue nodes as well as purple nodes. In a similar fashion, groups 7, 11, 14, 19, 23, 27, and 30 all split as well into three distinct groups. Groups 33 to 36 are introduced as well to account for blue nodes being present.

Table 5 Expansion of Node Groups When Unaligned (Blue) Nodes are Present.

Node Group	Node Color	Incident Edges	Symbol
1	Purple	None	(P:0)
2	Purple	Purple	(P:P)
3	Purple	pBp	(P:pBp)
4	Purple	pBb	(P:pBb)
5	Purple	pBp, pBb	(P:pBp/pBb)
6	Purple	pRp	(P:pRp)
7	Purple	Purple, pBp	(P:P/pBp)
8	Purple	Purple, pBb	(P:P/pBb)
9	Purple	Purple, pBp, pBb	(P:P/pBp/pBb)
10	Purple	Purple, pRp	(P:P/pRp)
11	Purple	pBp, pRp	(P:pBp/pRp)
12	Purple	pBb, pRp	(P:pBb/pRp)
13	Purple	pBp, pBb, pRp	(P:pBp/pBb/pRp)
14	Purple	Purple, pBp, pRp	(P:P/pBp/pRp)
15	Purple	Purple, pBb, pRp	(P:P/pBb/pRp)
16	Purple	Purple, pBp, pBb, pRp	(P:P/pBp/pBb/pRp)
17	Purple	pRr	(P:pRr)
18	Purple	Purple, pRr	(P:P/pRr)
19	Purple	pBp, pRr	(P:pBp/pRr)
20	Purple	pBb, pRr	(P:pBb/pRr)
21	Purple	pBp, pBb, pRr	(P:pBp/pBb/pRr)
22	Purple	pRp, pRr	(P:pRp/pRr)
23	Purple	Purple, pBp, pRr	(P:P/pBp/pRr)
24	Purple	Purple, pBb, pRr	(P:P/pBb/pRr)
25	Purple	Purple, pBp, pBb, pRr	(P:P/pBp/pBb/pRr)
26	Purple	Purple, pRp, pRr	(P:P/pRp/pRr)
27	Purple	pBp, pRp, pRr	(P:pBp/pRp/pRr)
28	Purple	pBb, pRp, pRr	(P:pBb/pRp/pRr)
29	Purple	Blue, pRp, pRr	(P:pBp/pBb/pRp/pRr)
30	Purple	Purple, pBp, pRp, pRr	(P:P/pBp/pRp/pRr)
31	Purple	Purple, pBb, pRp, pRr	(P:P/pBb/pRp/pRr)
32	Purple	Purple, pBp, pBb, pRp, pRr	(P:P/pBp/pBb/pRp/pRr)
33	Blue	pBb	(B:pBb)
34	Blue	bBb	(B:bBb)
35	Blue	pBb, bBb	(B:pBb/bBb)
36	Blue	None	(B:0)
37	Red	pRr	(R:pRr)
38	Red	rRr	(R:rRr)
39	Red	pRr, rRr	(R:pRr/rRr)
40	Red	None	(R:0)

Jaccard Similarity With Blue Nodes

Again, for simplicity, the main text only discussed the definition of our Jaccard Similarity (JS) score when there are no blue nodes present in the alignment. When blue nodes are not allowed, then for every node in G_1 , we can find (using the correct alignment) where that node is supposed to go in G_2 , and (using the given alignment) where it actually ends up in G_2 . These two nodes in G_2 can then be compared to create the JS score for the node.

However, when blue nodes are allowed, there are four possible cases that can arise instead of one:

- 1 The node is supposed to be aligned, and it is (the case described above) (“purple node stays purple”)
- 2 The node is supposed to be aligned, and it is not (“purple node turns to blue”)
- 3 The node is not supposed to be aligned, and it is (“blue node turn to purple”)
- 4 The node is not supposed to be aligned, and it is not (“blue node stays blue”)

VISNAB handles case 4 by simply assigning a score of 1.0 to the node, since it is correctly left unaligned. To deal with cases 2 and 3, VISNAB instead compares

two nodes in network G_1 . Specifically, for case 2, if a node a in G_1 is supposed to (using the correct alignment) be aligned to node n in G_2 , but is instead unaligned, we look to see which node b in G_1 is aligned (using the given alignment) to node n in G_2 . We then create the JS score for node a by comparing the neighborhoods of a and b in G_1 . If there is no node b (when nothing is aligned to node n in G_2 , i.e. it is “red”), then the JS score for node a is 0.0. Case 3 is handled analogously, again comparing two nodes a and b in G_1 to obtain a JS score, with a 0.0 assigned if there is no matching node in G_1 .

For some network $G = (V, E)$, let $N_G(z_1) = \{z_2 \in V : (z_1, z_2) \in E\}$ be the neighborhood of node z_1 in G . For nodes $x, y \in V$, let $N_G(x, y)$ be the neighborhood of x disregarding y , and let i_{xy} be a corrective term accounting for a possible edge between the two. Accordingly, if $y \in N_G(x)$ then $N_G(x, y) = N_G(x) - y$ and $i_{xy} = 1$, else $N_G(x, y) = N_G(x)$ and $i_{xy} = 0$. Let $N_G(y, x)$ be defined analogously. Our extended JS definition $\sigma_G : V \times V \rightarrow [0, 1]$ between two nodes is defined as:

$$\sigma_G(x, y) = \frac{|N_G(x, y) \cap N_G(y, x)| + i_{xy}}{|N_G(x, y) \cup N_G(y, x)| + i_{xy}}$$

Note that when x and y are both singletons, we define $\sigma(x, y) = 1.0$ to avoid dividing by zero.

$$f_a(u) = \begin{cases} \sigma_{G_2}(a(u), a_c(u)) & \text{if } a(u) \text{ and } a_c(u) \text{ are defined} \\ \sigma_{G_1}(u, a_c^{-1}(a(u))) & \text{if } a(u) \text{ is defined} \\ \sigma_{G_1}(u, a^{-1}(a_c(u))) & \text{if } a_c(u) \text{ is defined} \\ 1.0 & \text{if } a(u) \text{ and } a_c(u) \text{ are undefined} \end{cases}$$

Given node sets $V_a, V_c \in V_1$, an alignment $a : V_a \rightarrow V_2$, and the correct alignment $a_c : V_c \rightarrow V_2$, our JS measure for the given alignment a , with respect to the correct alignment a_c , is defined as:

$$JS(a) = \frac{1}{|V_1|} \sum_{u \in V_1} f_a(u)$$

If the correct alignment is provided, the user can choose to have correctly and incorrectly aligned nodes laid out separately in different node groups. The user can choose the criterion for the correct alignment to be based either on the traditional NC measure, or on our JS measure. If JS is chosen, the user can set the threshold value $\beta \in [0, 1]$, so, a node in the form of $u::v$ or $u::$ is denoted correct if $f_a(u) \geq \beta$.

Creation of the Correct Network Alignment

To create the “correct” alignment used in the case studies, we wanted to create two networks where all nodes in the smaller network had one and only one known matching node in the larger network. One consequence of this approach is that our correct alignment did not have any blue nodes. These case studies use two different protein-protein interaction datasets. The larger “SC” network, from *S. cerevisiae*, contains 5,831 nodes and 77,149 edges, and was originally obtained from BioGRID

(v3.2.101, June 2013) [14]. The smaller “Yeast2” network, also from *S. cerevisiae*, has 2,390 nodes and 16,127 edges. It was originally generated from data in [13] and used in [4]. Both networks were previously used in [16].

Nodes in Yeast2 are tagged with a variety of gene symbols (e.g. *PSY4*), secondary identifiers, and synonyms, while nodes in SC were tagged with ENTREZ IDs (e.g. 852234). In order to generate the “correct” alignment file, it was necessary to find the mapping from the former to the latter. To do this, we first used the YeastMine API [25, 26] at <https://yeastmine.yeastgenome.org/>, provided by the Saccharomyces Genome Database (SGD) [23], in order to generate a mapping from the node names to the SGD IDs that we could then feed to the DAVID web tool [27, 28]. With a Java program employing libraries provided by `org.intermine`, we downloaded (11 Feb. 2018) tuples for `Gene.primaryIdentifier`, `Gene.secondaryIdentifier`, `Gene.symbol`, and `Gene.synonyms.value` for `Gene.organism.shortName=‘S. cerevisiae’`, for all entries in the lists `Verified_ORFs`, `Dubious_ORFs` and `ALL_Verified_Uncharacterized_Dubious_ORFs`. Three remaining genes *YAR010C*, *YBR012W-B*, and *YHL009W-B* were not in any of these lists and were explicitly queried.

For each gene in Yeast2, we then matched the node name to a `Gene.synonyms.value`, and from this obtained a list of one or more `Gene.primaryIdentifiers`. In the cases where there was more than one, we chose the `Gene.primaryIdentifier` that mapped to a `Gene.symbol` that matched the `Gene.synonyms.value`. For example, synonym *MSL1* mapped to SGD IDs S000004374 and S000001448. However, while the former SGD ID mapped to gene symbol *NAM2*, the latter mapped to *MSL1*, and thus was selected. With one exception, this approach resulted in an unambiguous mapping of all Yeast2 node names to SGD IDs. The exception was for gene names *EFG1* and *YGR272C*; the latter was merged into the former, giving both names the same SGD ID (S000007608). Thus, node *YGR272C* was dropped (see Table 6)

Table 6 Node that was merged, and dropped

Gene Name	SGD ID
YGR272C	S000007608

These SGD IDs were then uploaded as a gene list to DAVID at:

<https://david.ncifcrf.gov/conversion.jsp>

(DAVID 6.8, accessed 18 Feb. 2018). Since DAVID has restrictions on large-scale queries through their web API, this was done manually. Upon uploading the list, DAVID’s Gene List Manager was not able to identify five IDs, so these nodes were dropped as well (see Table 7).

Table 7 Nodes not identified by DAVID, and dropped

Gene Name	SGD ID
IMD1	S000000095
YNL276C	S000005220
YDR133C	S000002540
YDL026W	S000002184
YAR075W	S000002145

We instructed the tool to convert SGD IDs to ENTREZ.GENE IDs, and downloaded the result. Thus, we had a mapping of 2,384 of the nodes in Yeast2 to

ENTREZ IDs. However, not all of these ENTREZ IDs are present as nodes in the larger SC network. In order to create a correct alignment with no blue nodes, we then pruned the Yeast2 network to remove the small number of nodes that could not be mapped onto the SC network. This resulted in an additional five nodes that needed to be dropped; see Table 8.

Table 8 Nodes that could not be mapped to SC network, and dropped

Gene Name	SGD ID
ATM1	855347
PHM8	856759
PUT1	850833
CTM1	856509
SBE2	851953

Detailed Description of the Node Assignment Algorithm for the Node and Link Group Layout

The node assignment algorithm for the Node and Link Group Layout is a multi-queue breadth first search graph traversal. While a typical breadth first search utilizes a single queue, our multi-queue approach uses one queue for each node group, and the queues are processed in the order listed in Table 2. The traversal starts on the node of highest degree in the first queue; its neighbor nodes are then visited in order of decreasing degree. If a newly visited node is in the current node group, it will be placed onto the current queue; if it is not, it will be placed onto the queue of its node group. The traversal is finished with a queue when every node in that node group has been visited. If the queue is empty but there still are unvisited nodes in that group, the highest degree node from the set of unvisited nodes of that group is added to the queue; after the queue is traversed, if there still are unvisited nodes in the group, this step is repeated until all nodes in the group are visited. Once finished, the traversal moves to the next queue. If a queue is empty when first evaluated, the node of highest degree in that queue's node group is added.

Thus, we created a “Yeast2-reduced” network consisting of 2,379 nodes and 16,063 edges, which was used in the case studies.

Full Table of All Alignment Scores for Mixtures of Importance and Symmetric Substructure Score

Table 9 lists the scores for the ten-hour SANA [16] runs between Yeast2K-Reduced and SC, in which we used combinations of Importance (I) [17] and Symmetric Substructure Score (S^3) [15] in the objective function. Note that all these scores, with the exception of Resnik, are available using the Alignment Measures tool in VISNAB. The Resnik scores [18, 19, 20] shown here are the means of the non-zero, non-“None” values computed separately using FastSemSim [21], incorporating Gene Ontology (GO) terms [29, 30] downloaded in February 2019.

Table of Node Group Sizes for Case III

Table 10 provides the number of nodes in each node group for the four alignments discussed in Case III. Asterisks show the four largest node groups per alignment, which are labeled prominently in Figure 5. As called out in the text, of the 716 nodes in the top 13 rows above group (**P:P/pRp/pRr**) for the mixed alignment, 544 (76%) have no incident **pRr** edges.

Table 9 All Alignment Scores for Case Study III

Alignment	NGS	LGS	NC	JS	S^3	Resnik
Correct	1.00	1.00	1.00	1.00	0.25	9.63
$1.0 * I$	0.61	0.79	0.00042	0.021	0.0043	3.16
$.001 * S^3 + .999 * I$	0.88	0.86	0.00042	0.024	0.17	3.48
$.003 * S^3 + .997 * I$	0.88	0.86	0.00042	0.021	0.18	3.39
$.005 * S^3 + .995 * I$	0.72	0.85	0.00	0.025	0.10	3.27
$.01 * S^3 + .99 * I$	0.88	0.86	0.00	0.024	0.19	3.32
$.03 * S^3 + .97 * I$	0.87	0.80	0.018	0.057	0.27	3.62
$.05 * S^3 + .95 * I$	0.73	0.53	0.022	0.063	0.49	3.44
$.1 * S^3 + .9 * I$	0.67	0.48	0.017	0.067	0.54	3.50
$1.0 * S^3$	0.64	0.46	0.021	0.069	0.55	3.61

Table 10 Sizes of all Node Groups in Case III

Symbol	Correct	All Importance	Mixed	All S^3
(P:0)	0	0	0	0
(P:P)	2	0	23	16
(P:pBp)	0	0	0	0
(P:pRp)	0	0	0	0
(P:P/pBp)	2	0	59	23
(P:P/pRp)	52	0	62	3
(P:pBp/pRp)	32	403*	207	0
(P:P/pBp/pRp)	27	9	193	1
(P:pRr)	0	0	0	0
(P:P/pRr)	5	0	31	439
(P:pBp/pRr)	5	35	98	303
(P:pRp/pRr)	0	0	0	0
(P:P/pBp/pRr)	1	9	43	662*
(P:P/pRp/pRr)	981*	0	530*	157
(P:pBp/pRp/pRr)	310	1677*	82	0
(P:P/pBp/pRp/pRr)	962*	246	1051*	775*
(R:pRr)	578*	843*	752 *	31
(R:rRr)	209	58	0	1087*
(R:pRr/rRr)	2665*	2551*	2700*	2334*
(R:0)	0	0	0	0

Percentage of Purple Nodes Without and With Incident **pRr** Edges between Correct and Mixed Alignments

The discussion of Figure 5 notes that while there are *more* **pRr** edges in the mixed alignment compared to the correct alignment, those edges are concentrated across a *smaller fraction* of the purple nodes in that mixed alignment. Table 11 compares the percentages of all purple nodes without [(P:*)] and with [(P:*/pRr)] **pRr** incident edges, between the correct and mixed alignments.

Table 11 Comparison of Purple Node Fractions Without and With **pRr** Edges

Alignment	(P:*)	(P:*/pRr)
Correct Alignment	4.83%	95.17%
Mixed Alignment	22.87%	77.13%

Alignment Cycle Layout With Blue Nodes

When unaligned blue nodes are not allowed, there are four cases that must be handled by the Alignment Cycle layout, and these are indicated by a checkmark in the rightmost column of Table 12. When unaligned blue nodes are present, there are nine path types that must be handled. In the table, a network with nodes $\{A, B, C, \dots, L\}$ has been aligned onto a network with nodes $\{1, 2, 3, \dots, 12\}$. Note that purple node runs can extend for any length of nodes, as shown by the , but the matches and alignments given in this table are for the cases where there

are none of these extra nodes. The Alignment Cycle layout will order the nodes in the path and cycle cases so that misaligned nodes are laid out next to their correct partners; see case 9 in particular to see this pattern.

Table 12 Alignment Cycle Layout Cases

#	Layout	Correct Alignment	Test Alignment	Type	No Blue
1	(A::)	$A \rightarrow \emptyset$	$A \rightarrow \emptyset$	Correct	
2	(::1)	$\emptyset \rightarrow 1$	$\emptyset \rightarrow 1$	Correct	✓
3	(B::2)	$B \leftrightarrow 2$	$B \rightarrow 2$	Correct	✓
4	(C::3)	$C \rightarrow \emptyset$ $\emptyset \rightarrow 3$	$C \rightarrow 3$	Path	
5	(::4) (D::)	$4 \leftrightarrow D$	$\emptyset \rightarrow 4$ $D \rightarrow \emptyset$	Path	
6	(::5) (E::6) ...	$5 \leftrightarrow E$ $\emptyset \rightarrow 6$	$\emptyset \rightarrow 5$ $E \rightarrow 6$	Path	✓
7	(F::7) ... (G::)	$7 \leftrightarrow G$ $F \rightarrow \emptyset$	$F \rightarrow 7$ $G \rightarrow \emptyset$	Path	
8	(::8) (H::9) ... (I::)	$8 \leftrightarrow H$ $9 \leftrightarrow I$	$\emptyset \rightarrow 8$ $H \rightarrow 9$ $I \rightarrow \emptyset$	Path	
9	(J::10) (K::11) ... (L::12)	$10 \leftrightarrow K$ $11 \leftrightarrow L$ $12 \leftrightarrow J$	$J \rightarrow 10$ $K \rightarrow 11$ $L \rightarrow 12$	Cycle	✓

The Four-Cluster Misalignment

In Case Study IV, we showed how the Alignment Cycle layout could be used to spot alignment problems such as two entire protein clusters being swapped. Figure 8 shows a severe degeneracy for the same alignment run, where *four* separate protein clusters were misaligned in a cycle. The BioFabric depiction of this problem follows the same pattern shown in Figure 7, but the successive edge wedges are even steeper here, and show a clear pattern of cycling between four distinct sets of node rows.

Availability and requirements

Project Name: VISNAB: Visualization of Network Alignments using BioFabric

Project Home Page <http://www.BioFabric.org/VISNAB/index.html>. The VISNAB code repository is at <https://github.com/wjrl/AlignmentPlugin>, while the BioFabric code repository is at <https://github.com/wjrl/BioFabric>.

Operating Systems Cross-platform.

Programming Language Java

Other Requirements Plugin requires BioFabric Version 2 Beta Release 2 or above, available on GitHub and the BioFabric project Home Page. An OpenJDK Java runtime is bundled with BioFabric and does not need to be installed separately.

License LGPL V 2.1. Per the LGPL license, the VISNAB source code is provided in additional file 2.

List of Abbreviations

EC: Edge Coverage, Conservation, Correctness, or Correspondence

ICS: Induced Conserved Structure

JS: Jaccard Similarity

LCES: Largest Common Connected Subgraph

LGS: Link Group Similarity

NC: Node Correctness

NGS: Node Group Similarity

PPI: protein-protein interaction

(S^3): Symmetric Substructure Score

SGD: Saccharomyces Genome Database

VISNAB: Visualization of Network Alignments using BioFabric

Author details

¹Department of Computer Science, University of California, Irvine CA, 92697-3435 USA. ²Institute for Systems Biology, Seattle WA, 98109-5234 USA.

References

1. Milano M, Guzzi PH, Tymofiyeva O, Xu D, Hess C, Veltri P, et al. Multiple Anonymized Social Networks Alignment. 2015 IEEE International Conference on Data Mining. 2015;p. 599–608.
2. Milano M, Tymofiyeva O, Xu D, Hess C, Cannataro M, Guzzi P. Using Network Alignment for Analysis of Connectomes: Experiences from a Clinical Dataset. In: Proceedings of the 7th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics. BCB'16. New York, NY, USA: ACM; 2016. p. 649–656. Available from: <https://dl.acm.org/citation.cfm?doid=2975167.2985690>.
3. Milano M, Guzzi P, Tymofiyeva O, Xu D, Hess C, Veltri P, et al. An extensive assessment of network alignment algorithms for comparison of brain connectomes. BMC Bioinformatics. 2017;18:235.
4. Kuchaiev O, Milenković T, Memišević V, Hayes W, Pržulj N. Topological network alignment uncovers biological function and phylogeny. Journal of The Royal Society Interface. 2010;7(50):1341–1354.
5. Uetz P, Dong YA, Zeretzke C, Atzler C, Baiker A, Berger B, et al. Herpesviral protein networks and their interaction with the human proteome. Science. 2006;311(5758):239–242.
6. Milenković T, Zhao H, Faisal FE. Global Network Alignment In The Context Of Aging. In: Proceedings of the International Conference on Bioinformatics, Computational Biology and Biomedical Informatics. BCB'13. New York, NY, USA: ACM; 2013. p. 23:23–23:32. Available from: <http://doi.acm.org/10.1145/2506583.2508968>.
7. Longabaugh WJR. Combing the hairball with BioFabric: a new approach for visualization of large networks. BMC Bioinformatics. 2012;13:275.
8. Tamassia R, Tollis IG. A unified approach to visibility representations of planar graphs. Discrete Comput Geom. 1986;1(4):321–341.
9. Blakley B. Reduction of Flow Diagrams to Unfolded Form Modulo Snarls. Defense Technical Information Center; 1987. to AFOSR on Contract F49620-86-C-0103.
10. McAllister AJ. A new heuristic algorithm for the linear arrangement problem. University of New Brunswick: Faculty of Computer Science; 1999. 99.126a.
11. Xie J, Xiang C, Zhou Z, Dai D, Zhang H. NetCompare: A visualization tool for network alignment on Galaxy. In: 2014 International Conference on Information Science, Electronics and Electrical Engineering. vol. 2; 2014. p. 881–884.
12. Malek M, Ibragimov R, Albrecht M, Baumbach J. CytoGEDEVO-global alignment of biological networks with Cytoscape. Bioinformatics. 2016;32(8):1259–1261.
13. Collins SR, Kemmeren P, Zhao XC, Greenblatt JF, Spencer F, Holstege FCP, et al. Toward a Comprehensive Atlas of the Physical Interactome of *Saccharomyces cerevisiae*. Molecular and Cellular Proteomics.

- 2007;6(3):439–450. Available from: <http://www.mcponline.org/content/6/3/439.abstract>.
14. Chatr-aryamontri A, Breitkreutz BJ, Heinicke S, Boucher L, Winter A, Stark C, et al. The BioGRID interaction database: 2013 update. *Nucleic Acids Research*. 2013;41(D1):D816–D823. Available from: <http://nar.oxfordjournals.org/content/41/D1/D816.abstract>.
 15. Saraph V, Milenković T. MAGNA: maximizing accuracy in global network alignment. *Bioinformatics*. 2014;30(20):2931–2940.
 16. Mamano N, Hayes WB. SANA: simulated annealing far outperforms many other search algorithms for biological network alignment. *Bioinformatics*. 2017;33(14):2156–2164.
 17. Hashemifar S, Xu J. HubAlign: an accurate and efficient method for global alignment of protein–protein interaction networks. *Bioinformatics*. 2014 Sep;30(17):i438–i444. Available from: <http://dx.doi.org/10.1093/bioinformatics/btu450>.
 18. Resnik P. Using Information Content to Evaluate Semantic Similarity in a Taxonomy. In: *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 1. IJCAI'95*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.; 1995. p. 448–453. Available from: <http://dl.acm.org/citation.cfm?id=1625855.1625914>.
 19. Lord PW, Stevens RD, Brass A, Goble CA. Semantic similarity measures as tools for exploring the gene ontology. In: *Proc. of the 8th Pacific Symposium on Biocomputing*; 2003. p. 601–612.
 20. Lord PW, Stevens RD, Brass A, Goble CA. Investigating semantic similarity measures across the Gene Ontology: the relationship between sequence and annotation. *Bioinformatics*. 2003;19(10):1275–1283. Available from: <http://dx.doi.org/10.1093/bioinformatics/btg153>.
 21. Guzzi PH, Mina M, Guerra C, Cannataro M. Semantic similarity analysis of protein data: assessment with biological features and issues. *Briefings in bioinformatics*. 2012;13(5):569–585.
 22. Shannon P, Markiel A, Ozier O, Baliga NS, Wang JT, Ramage D, et al. Cytoscape: A Software Environment for Integrated Models of Biomolecular Interaction Networks. *Genome Research*. 2003;13(11):2498–2504.
 23. Cherry J, Adler C, Ball C, Chervitz S, Dwight S, Hester E, et al. SGD: *Saccharomyces Genome Database*. *Nucleic Acids Research*. 1998;26(1):73–79.
 24. Vijayan V, Milenković T. Multiple network alignment via multiMAGNA++. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*. 2018;p. 1–1.
 25. Balakrishnan R, Park J, Karra K, Hitz BC, Binkley G, Hong EL, et al. YeastMine—an integrated data warehouse for *Saccharomyces cerevisiae* data as a multipurpose tool-kit. *Database*. 2012;2012:bar062. Available from: <http://dx.doi.org/10.1093/database/bar062>.
 26. Smith RN, Aleksic J, Butano D, Carr A, Contrino S, Hu F, et al. InterMine: a flexible data warehouse system for the integration and analysis of heterogeneous biological data. *Bioinformatics*. 2012;28(23):3163–3165. Available from: <http://dx.doi.org/10.1093/bioinformatics/bts577>.
 27. Huang DW, Sherman BT, Lempicki RA. Bioinformatics enrichment tools: paths toward the comprehensive functional analysis of large gene lists. *Nucleic Acids Res*. 2009 Jan;37(1):1–13. Gkn923[PII]. Available from: <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2615629/>.
 28. Huang DW, Sherman BT, Lempicki RA. Systematic and integrative analysis of large gene lists using DAVID bioinformatics resources. *Nature Protocols*. 2008 Dec;4:44 EP –. Available from: <http://dx.doi.org/10.1038/nprot.2008.211>.
 29. Ashburner M, Ball CA, Blake JA, Botstein D, Butler H, Cherry JM, et al. Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. *Nat Genet*. 2000;25:25–9.
 30. Consortium TGO. The Gene Ontology Resource: 20 years and still GOing strong. *Nucleic Acids Res*. 2019;47:D330–D338.

Additional Files

These files are available at: <https://www.ics.uci.edu/~wayne/papers/SpringerNetworkChapter2020/>

Additional file 1: DesaiEtAl-2019-CaseStudyFiles.zip

Archive of .sif, .gw, .align, .resnik, and .bif (BioFabric) files for the four case studies. The latter can be loaded into BioFabric with VISNAB to view the case study results in detail.

Additional file 2: VISNAB-1.1.0.0-src.tar.gz

Contains the source code for the VISNAB V1.1 plugin. The most recent source code is available at the GitHub repository listed above.

Additional file 3: BioFabric-2.0.B.2-src.tar.gz

Contains the source code for the beta version of BioFabric 2.0 needed to run the plugin. The most recent recent source code is available at the GitHub repository listed above.

Additional file 4: sVISNAB-V1.1.0.0.jar

The plugin jar for VISNAB V1.1.

Additional file 5: BioFabric-2.0.B.2.tgz

Contains a Linux executable for a beta version of BioFabric 2.0 needed to run the plugin. Executables for Windows and Mac are available on GitHub. In a pinch, the bundled OpenJDK for Linux in this provided executable can be swapped out for other versions to run the program on Windows or Mac from a batch file or shell script.