Fast and High Quality Graph Alignment via Treelets

Morgan Lee and George M. Slota
Rensselaer Polytechnic Institute, Troy, NY
leeh17@rpi.edu, slotag@rpi.edu

Abstract—**This work is a preliminary validation on using *treelets* in lieu of *graphlets* for the alignment of biological interaction networks. There has been considerable prior work on calculating alignments from vertex-based topological similarity measures based on local subgraph (i.e., graphlet) counts. In this work, we instead consider *treelets*, which are small acyclic subgraphs. As tree subgraphs can be enumerated quite quickly in parallel, we are able to consider larger subgraphs to better capture topological similarities in graph regions. We demonstrate that relative to the prior work using graphlets, our approach is more scalable and outputs similar or higher quality alignments.**

Index Terms—**graph alignment; subgraph isomorphism; treelets**

I. Introduction

The network alignment problem can be loosely defined as determining an injective vertex-to-vertex or edge-to-edge mapping between two disparate networks that minimizes some cost function [3]. The alignment problem can be considered a variation of the graph or subgraph isomorphism problem. Similar to the subgraph isomorphism problem, determining alignments between networks can often reveal latent functional similarities between the network structures, such as the functional role of proteins [9], [10]. Network alignments can be cheaper to calculate than exact subgraph matches (depending on the algorithm and subgraph size), so determining alignments between two very large networks is feasible.

Many algorithms exist for the graph alignment problem [2], [7], [9]. These generally calculate some local topologically-based feature vector, which can then be used to compute cost metrics between vertices and edges in disparate networks. Algorithms then attempt to minimize such a cost over a full pairwise alignment.

We specifically consider an alignment approach which utilizes the relative frequency of *graphlets* rooted on each vertex to define a similarity metric between vertices of separate networks [6], [10], [11], [13]. Graphlets are formally defined as all possible simple, undirected, and induced subgraphs from 2-5 vertices [16]. A vertex-to-vertex similarity score can be calculated based on *graphlet degree distributions*, where the *graphlet degree* of a vertex is the number of graphlet embeddings rooted at that vertex [15]. By taking a single vertex and calculating all possible graphlet degrees rooted at that vertex, a feature vector can be constructed (*graphlet degree signature*) for pairwise vertex similarity comparisons [14]. By examining the graphlet degree signatures for nodes within and between two distinct networks, an alignment between the networks can be determined.

The primary challenge of using graphlet counts is a computational one: the subgraph enumeration problem requires $O(n^k)$ work, where n is the graph order and k is the subgraph order. Enumerating all graphlets up to $k = 5$ correspondingly requires $O(n^5)$ for naïve methods or $O(n \cdot d^4)$ for optimized ones [8] (d is the maximum vertex degree) – infeasible for large-scale networks. In this work, we overcome this challenge by instead considering only acyclic (tree) subgraphs. Using the color-coding technique of Alon et al. [1], tree subgraphs can be approximately (but with very low error [20]) enumerated in $O(e^k m)$; i.e,. scaling linear in the size of a network (m) for a fixed k. This allows us to consider a both a greater number of subgraphs as well as subgraphs of a larger size when calculating a graphlet degree feature vector. As color-coding uses a dynamic programming approach, memory consumption might be larger than graphlet enumeration methods. However, this only becomes problematic on a typical HPC compute node for graphs of billions of edges and subgraphs of a dozen or mode vertices [20]; much larger than the current scale being considered.

We utilize a fast parallel tree enumeration tool (FASCIA [19], [20]) and our new parallel and scalable implementations of various alignment methods [6], [10] to calculate the feature vector and perform alignment. We demonstrate that the use of tree subgraphs can have a noted benefit on alignment quality, and that the use of tree subgraphs can also enable future scalability of subgraph-based alignment methods to network of considerably greater scale. We term our new approach as FASTALIGN. [1]

II. Background

A. Graphlets

Graphlets are formally defined as small undirected induced subgraphs between two and five vertices in size. Prior work by Pržulj et al. [14]–[17] extensively studies graphlets in the context of biological network analysis. Pržulj et al. also identified all possible discrete *orbits* within each graphlet. Orbits in this context refer to distinct automorphic vertices in the subgraph; i.e., it can be useful to explicitly differentiate between an embedding rooted on a vertex in the center of a star versus a vertex on one of the leaves, while we don't want to differentiate between an embedding rooted on "different" leaves. Solava et al. [23] extends this concept to edges, where graphlet counts are rooted based on distinct edge orbits instead.

[1] This work is an extension of that appearing in a prior dissertation [22].

1) Graphlet Degree Signature Similarity: The graphlet degree signature similarity is a per-vertex score that allows comparison between two disparate vertices in the same or separate networks. This score is based on a feature vector created with the counts for all possible graphlet orbits rooted at a given vertex. It is described as capturing the local topology and interconnectedness of the node in the context of its local neighborhood [14]. A distance value between two vertices, u and v, for graphlet orbit i with counts u_i and v_i is calculated as follows:

$$D_i(u,v) = 1 - w_i \times \frac{|\log(u_i+1) - \log(v_i+1)|}{\log(\max\{u_i, v_i\}+2)}$$

In this equation, w_i is a weighting given to each specific orbit. These values are dependent on the number of isomorphisms of smaller graphlets rooted at at orbit i. The total distance value between vertices is the sum of distance values divided by the sum of the weightings for all orbits.

$$D(u,v) = \frac{\sum_i D_i(u,v)}{\sum_i w_i}$$

The similarity between two vertices is then $S(u,v) = 1 - D(u,v)$. These above distance and similarity measures can be directly applied to edge-based orbits. To calculate D_i and D for edges we would instead consider replacing vertices (u,v) with edges (e,f) and consider the counts of e_i to be the number of times edge e has a rooted embedding of orbit i.

B. Treelets

Treelets in the context of this work are formally defined as all possible 3-7 vertex tree-structured non-induced subgraphs. Previous work [21] has demonstrated the applicability of using treelets in lieu of graphlets to benefit from the much lower possible running time bounds. There have been sampling methods introduced to improve time to solution for graphlets, but these still do not improve upon the upper bound, can still have relatively high cost, and have not yet been demonstrated in practice as applicable for enumerating per-vertex counts, only global counts.

It has been previously demonstrated [21] that treelet counts are generally more computationally efficient to compute and can take the place of graphlet counts for a number of the proposed graphlet-based analytics [15], [17]. The primary focus of this work is to demonstrate the possibility of using a *treelet degree signature* vector, calculated using the above equations, as a means to align biological networks.

III. Methods

A. GRAAL

The GRAAL (GRAph ALignment) algorithm and its variants [10]–[13] constitute various approaches for the use of graphlet degree signature similarity between vertices of different networks to compute an alignment. A brief overview of the baseline GRAAL algorithm implemented in this work is given by Algorithm 1 (see [10] for more details). The algorithm proceeds as follows. First, a cost matrix C is created between all possible vertices v and u in between the two networks G and H based on the following function:

$$C(u,v) = 2 - \Big((1-\alpha) \times \frac{v_d + u_d}{\max_d(G) + \max_d(H)} + \alpha \times S(u,v) \Big)$$

In this function, v_d and u_d are the degrees of vertex v and u, $\max_d(G)$ and $\max_d(H)$ are the maximum degrees of graphs G and H, $S(v,u)$ is the graphlet degree signature similarity between v and u, and α is a control parameter between $[0,1]$ that varies the influence of the vertex degrees versus signature similarity on the overall cost. An α value of 0 would specify that only vertex degrees are to be utilized, while an α value of 1 would result in only vertex counts being utilized.

Algorithm 1 GRAAL Alignment Algorithm

```
procedure GRAAL($G, H$)
    $C \leftarrow$ allCosts($G, H$) in parallel
    $A \leftarrow \varnothing$
    $p \leftarrow 1$
    while $G, H$ not fully aligned do
        $(u, v) \leftarrow$ findSeed($G^p, H^p$) in parallel
        $A \leftarrow (u, v)$
        $r \leftarrow 1$
        repeat
            $R_G \leftarrow$ getRadius($G^p, u, r$) in parallel
            $R_H \leftarrow$ getRadius($H^p, v, r$) in parallel
            $A \leftarrow$ align($R_G, R_H, C$)
            $r \leftarrow r + 1$
        until $R_G$ or $R_H = \varnothing$
        if $r \geq 3$ and $p < 3$ then
            $p \leftarrow p + 1$
return $A$
```

Using the cost matrix, an initial *seed* is selected as the minimal value pair in C and added to the alignment A. The networks are then iteratively and greedily aligned (based on minimal cost) outward from this pair of vertices on a per-radius basis (e.g. 1 hop from u and v, 2 hops, 3 hops, etc.) until no more vertices are available for alignment in one of the graphs at a given radius.

If the resultant radius is greater than or equal to 3, the graphs are taken to the next *power*. In this instance, power refers to a graph that is created by adding edges to the graph between all vertices having some shortest paths length between them up to some value in the original graph. For example, a power 1 graph would just be the original graph (i.e. $G^1 = G$), while a power 2 graph would have additional edges between all vertices that are at most 2 hops away from each other on the original graph. This allows for inexactness in the alignment, similar to how additions or deletions function in sequence alignment.

New seeds are selected and the iterative alignment procedure is again performed for each radius. This continues with new seeds being selected and the graphs incremented as necessary up to a power as 3 (G^3) until all possible vertices in the smaller of G and H have been fully aligned.

B. GREAT

The GREAT (GRaphlet Edge-based AlignmenT) algorithm [6] uses edge-based similarity measures in addition to the vertex-based measures as used in GRAAL. It first considers defining alignment score based on edge orbits as:

$$C_e(e, f) = (1 - \alpha) \times \frac{e_d + f_d}{\max_d(G) + \max_d(H)} + \alpha \times S_e(e, f)$$

where e_d is a measure of *edge degree centrality* ($e_d = \sum_i w_i \times \log(e_i + 1)$, for graphlets i with weight w_i and count per edge e_i, similar as before), S_e is the edge-based graphlet degree signature similarity noted in Section II-A1, and $\max_d(G)$ refers to the maximum edge degree centrality over all edges in G. GREAT then uses these costs to calculate a greedy edge-to-edge pairwise alignment between G and H. This alignment is then used to create pairwise vertex similarities $sim(u, v)$ by summing up the alignment scores of all edges aligned in both of the neighborhoods of u and v. These scores can then be directly input into a vertex aligner such as GRAAL.

C. Parallelization

We provide novel parallelization for a number of subroutines within GRAAL and GREAT. Within GRAAL, we use `OpenMP` to parallelize creation of the cost matrix, seed search, as well as the graphs and subgraphs created when increasing the graph power and determining the local radius. We also parallelize the creation of the cost matrix for GREAT and can parallelize the determination of the minimum cost edge pairs when doing the greedy alignment. Alignment itself is non-trivial to parallelize due to strong dependencies, but exploring techniques to enable parallelization is an interesting avenue for future work.

D. Treelet and Graphlet Counting

We use the FASCIA [20] parallel treelet counting tool to extract per-vertex treelet counts. We modify the program to enable the output of vertex-based counts for GRAAL and edge-based counts for use with GREAT. We retrieve graphlet counts using Orca [8]. A number of other graphlet counting tools exist, however, Orca was the fastest such tool we could find that counted up to 5-vertex graphlets and output the per-vertex counts we need to compute similarity measures.

E. Alignment Evaluation

There are four metrics commonly used to evaluate the quality of alignment between two biological networks. These are edge correctness, symmetric substructure score (S^3), node correctness and interaction correctness. Given an alignment in terms of a mapping function M from vertex sets $V_G \in G$ to $V_H \in H$, we can define symmetric substructure score as the following:

$$|E_G \cap E_H| = |(u, v) \in E_G : (M(u), M(v)) \in E_H|$$

$$S^3 = \frac{|E_G \cap E_H|}{|E_H| + |E_G| - |E_G \cap E_H|}$$

where E_G and E_H are the edge sets of G and H, respectively, and $M(u)$ defines the vertex that v is mapped to in H. This metric can be simply stated as the ratio of the number of edges that exist in G that equivalently end up mapped to an existing edge in H over the total number of edges in G and H minus the number mapped. Edge correctness is similar to the S^3 score, but is known to have certain drawbacks [6]. Node correctness and interaction correctness can evaluate alignments on labeled protein interaction datasets, where protein labels and interactions are known. We only consider unlabeled data in this exploratory work.

IV. RESULTS

Experiments were performed on dual socket Xeon(R) Platinum 8160 CPU node with 196 GB DDR. We use biological interaction networks for testing, with the datasets retrieved from several sources [4], [5], [24]. We use protein interaction networks for humans and yeast, and we include a *C. elegans* gene interaction network for larger scale performance tests. We compare three alignment methods: a baseline with GRAAL using graphlet counts (Graphlets), GRAAL using treelet counts (Treelets), and GREAT using treelet edge counts (TreeletEdges). We run with 100 iterations of FASCIA in all tests, unless otherwise noted.

A. Alignment Quality

We first compare alignment quality, given in Figure 1. We run on the Yeast network, with 5%, 10%, 15%, and 20% edges rewired, and we compare the S^3 alignment score versus varying α. On average, the use of treelets improves alignment quality by 3.1% over graphlets, while the use of the additional edge alignment information improves quality by 9.2% on average over the baseline.

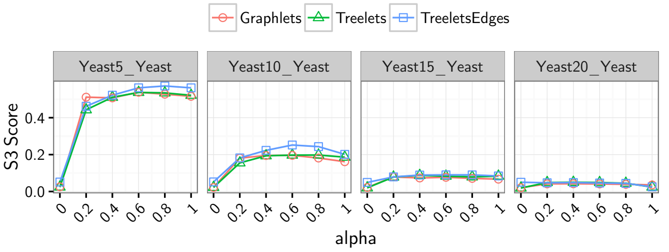

Fig. 1. S^3 score versus α for our three tested alignment methods.

We also consider the effects of iteration counts for FASCIA in determining alignment quality. We run on 1 to 10,000 iterations using just GRAAL and align the Human network and the Yeast network, with the results versus α given in Figure 2. We note that even when using a low number of iterations, alignment quality is minimally effected. For our experiments in Figures 1 and 2, we used 100 iterations, though this could very easily have been lowered while still retaining good accuracy.

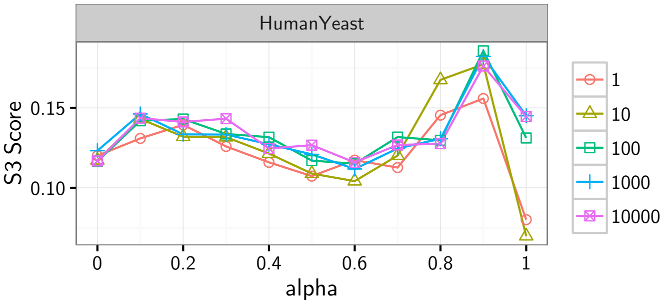

Fig. 2. S^3 score versus α and number of iterations of FASCIA.

TABLE I
GRAPHLET/TREELET COUNTING TIMES FOR THE TEST NETWORKS.

Network	n	m	Orca	FASCIA	Source
Yeast	5.1 K	22 K	4.1s	11s	[24]
Human	9.1 K	41 K	9.1s	18s	[18]
C.elegans	15 K	246 K	777s	51s	[4]

B. Subgraph Count Execution Times

We finally compare the execution time for counting Treelets and Graphlets in Figure I. As noted, we use 100 iterations for FASCIA (Treelets) and compare to Orca (Graphlets) on the three considered networks. The difference in the alignment phase computational costs is minimal between the two methods, as the only difference is in initializing the cost matrix.

We note that the difference in compute time becomes considerable as the graphs increase in scale. The time for FASCIA are also relatively conservative, as we can likely decrease the iteration count by an order-of-magnitude without much impact on alignment quality. The performance difference we show here between graphlets and treelets is a key consideration if applying subgraph-based alignments to larger-scale networks than considered in this paper, as the scaling behavior of Treelet counting is considerably more favorable.

V. DISCUSSION

This paper considered a preliminary application of *treelets* in lieu of *graphlets* for the purpose of biological network alignment. We implemented and parallelized the GRAAL subgraph-based graph alignment algorithm, and demonstrated that use of treelets has the potential to enable considerable scalability for subgraph-based alignments with an additional noted benefit to alignment quality. The improvements in alignment quality might be explained by the usage of larger-sized treelets, which can capture the topological signature of an extended per-vertex neighborhood relative to graphlets. We also extend our preliminary work by considering edge-alignment via GREAT, and we observe additional improvement to alignment quality over our baseline. Future work will further investigate the scaling of these techniques to larger graphs, incorporating techniques from the multitude of other subgraph-based alignment strategies [11], [13] to improve overall alignment quality, and comparing against other general alignment methods [2].

REFERENCES

[1] N. Alon, R. Yuster, and U. Zwick, “Color-coding,” *J. ACM*, vol. 42, no. 4, pp. 844–856, 1995.

[2] M. Bayati, D. F. Gleich, A. Saberi, and Y. Wang, “Message-passing algorithms for sparse network alignment,” *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 7, no. 1, pp. 1–31, 2013.

[3] J. Berg and M. Lässig, “Local graph alignment and motif search in biological networks,” *Proceedings of the National Academy of Sciences*, vol. 101, no. 41, pp. 14 689–14 694, 2004.

[4] A. Cho, J. Shin, S. Hwang, C. Kim, H. Shim, H. Kim, H. Kim, and I. Lee, “Wormnet v3: a network-assisted hypothesis-generating server for caenorhabditis elegans,” *Nucleic acids research*, vol. 42, no. W1, pp. W76–W82, 2014.

[5] S. R. Collins, P. Kemmeren, X.-C. Zhao, J. F. Greenblatt, F. Spencer, F. C. Holstege, J. S. Weissman, and N. J. Krogan, “Toward a comprehensive atlas of the physical interactome of saccharomyces cerevisiae,” *Molecular & Cellular Proteomics*, vol. 6, no. 3, pp. 439–450, 2007.

[6] J. Crawford and T. Milenković, “Great: graphlet edge-based network alignment,” in *2015 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE, 2015, pp. 220–227.

[7] M. Heimann, H. Shen, T. Safavi, and D. Koutra, “Regal: Representation learning-based graph alignment,” in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 2018, pp. 117–126.

[8] T. Hočevar and J. Demšar, “A combinatorial approach to graphlet counting,” *Bioinformatics*, vol. 30, no. 4, pp. 559–565, 2014.

[9] B. P. Kelley, R. Sharan, R. M. Karp, T. Sittler, D. E. Root, B. R. Stockwell, and T. Ideker, “Conserved pathways within bacteria and yeast as revealed by global protein network alignment,” *Proceedings of the National Academy of Sciences*, vol. 100, no. 20, pp. 11 394–11 399, 2003.

[10] O. Kuchaiev, T. Milenkovič, V. Memisević, W. Hayes, and N. Pržulj, “Topological network alignment uncovers biological function and phylogeny,” *Journal of the Royal Society Interface*, 2010.

[11] O. Kuchaiev and N. Pržulj, “Integrative network alignment reveals large regions of global network similarity in yeast and human,” *Bioinformatics*, 2011.

[12] V. Memisević and N. Pržulj, “C-GRAAL: common-neighbors-based global GRAph ALignment of biological networks,” *Integrative Biology*, 2012.

[13] T. Milenkovič, W. L. Ng, W. Hayes, and N. Pržulj, “Optimal network alignment with graphlet degree vectors,” *Cancer Informatics*, 2010.

[14] T. Milenkovič and N. Pržulj, “Uncovering biological network function via graphlet degree signatures,” *Cancer Informatics*, vol. 6, pp. 257–273, 2008.

[15] N. Pržulj, “Biological network comparison using graphlet degree distribution,” *Bioinformatics*, vol. 23, no. 2, pp. e177–83, 2007.

[16] N. Pržulj, D. G. Corneil, and I. Jurisca, “Modeling interactome, scale-free or geometric?” *Bioinformatics*, vol. 20, no. 18, pp. 3508–3515, 2004.

[17] N. Pržulj, D. Corneil, and I. Jurisica, “Efficient estimation of graphlet frequency distributions in protein-protein interaction networks,” *Bioinformatics*, vol. 22, no. 8, pp. 974–980, 2006.

[18] P. Radivojac, K. Page, W. T. Clark, B. J. Peters, A. Mohan, S. M. Boyle, and S. D. Mooney, “An integrated approach to inferring gene-disaese assicoations in humans,” *Proteins*, 2008.

[19] G. M. Slota and K. Madduri, “Parallel color-coding,” *Parallel Computing, Systems & Applications*, vol. 47, pp. 51–69, August 2015.

[20] ——, “Fast approximate subgraph counting and enumeration,” in *2013 International Conference on Parallel Processing (ICPP13)*, 2013.

[21] ——, “Complex network analysis using parallel approximate motif counting,” in *28th IEEE International Parallel and Distributed Processing Symposium (IPDPS14)*, 2014.

[22] G. M. Slota, “Irregular graph algorithms on modern multicore, manycore, and distributed processing systems,” *PhD thesis*, 2016.

[23] R. W. Solava, R. P. Michaels, and T. Milenković, “Graphlet-based edge clustering reveals pathogen-interacting proteins,” *Bioinformatics*, vol. 28, no. 18, pp. i480–i486, 2012.

[24] I. Xenarios, L. Salwinski, X. J. Duan, P. Higney, S. M. Kim, and D. Eisenberg, “DIP, the database of interacting proteins: a research tool for studying cellular networks of protein interactions,” *Nucleic Acids Research*, vol. 30, no. 1, pp. 303–305, 2002.