Fast and High Quality Graph Alignment
via Treelets

Morgan Lee and George M. Slota

Rensselaer Polytechnic Institute

HiCOMB 2020
**Basic definition:** Determining a pairwise vertex-to-vertex mapping between two graphs (\( H \rightarrow G \)) that minimizes some cost function. This is similar to subgraph isomorphism, but we allow some “error” or inexactness in the isomorphic relation.
Such an alignment can reveal functional similarities between biological interaction networks. Using graph alignment as a tool for biological network analytics has:

- Found consistent protein interaction network topologies across species as distinct as yeast and human [Kuchaiev et al., 2010].

- Predicted protein interactions not previously measured using this topological similarity [Malod-Dognin and Pržulj, 2015].

- Been a means to study the phylogenetics of various herpes viruses [Kuchaiev and Pržulj, 2011].
Graph Alignment: How

One approach is define a per-vertex feature vector consisting of counts of various subgraphs and minimizes the differences in these feature vectors when mapping vertices.\(^1\)

- Consider aligning network \(H\) to network \(G\).
- We count how often some number of distinct subgraphs are rooted at all \(u \in V(H)\) and \(v \in V(G)\).
- We define a cost of aligning each \(u\) to each \(v\).
- We attempt to minimize this cost over an entire alignment.

\(^1\)[Kuchaiev et al., 2010]
Consider the embedding frequency of various subgraphs to define a feature vector defining the local topology of some vertex \( v \). Intuitively, vertices in separate networks that have a similar local topology would make good candidates for some alignment mapping.
Define a per-subgraph distance between some vertex \( u \in V(H) \) and \( v \in V(G) \) based on the counts of subgraph \( i \) rooted on \( u \) and \( v \).

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

The total distance between \( u \) to \( v \) is the sum of each subgraph distance along with a per-subgraph weighting term \( w_i \).

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

Then the total cost of mapping \( u \) to \( v \) is a function of this distance, their degrees \( d(u) \) and \( d(v) \), the maximum degrees in the networks of \( \Delta(G) \) and \( \Delta(H) \), and tuning parameter \( \alpha \).

\[
C(u, v) = 2 - \left( (1 - \alpha) \times \frac{d(v) + d(u)}{\Delta(G) + \Delta(H)} + \alpha \times (1 - D(u, v)) \right)
\]

A greedy approach minimizes these cost over some pairwise mapping.
An overview iterative and greedy approach is as follows:

- Select the minimum $u, v$ over all $C(u, v)$ and align $u \rightarrow v$.
- Greedily align the $k$-hop neighborhoods of $u$ and $v$.
- Once the neighborhoods are full aligned, raise the graph to the next power – add edges between all vertices within 2-hops of each other.
- Repeat the above process until all $u \in V(H)$ is aligned.

By raising the graph to some $p^{th}$ power, we allow for inexact alignments, such as with gaps in Smith-Waterman sequence alignment. Our insertions and deletions, however, are in terms of missing and extra edges between the two networks.
Also Possible: The Use of Edge-based Counts

Subgraphs can also be considered rooted on a given edge $e$ instead of a vertex. A similar greedy algorithm can be constructed using this notion\(^2\).

\[ \begin{align*}
\text{e's neighborhood} &= 5 \\
\text{embedding frequencies} &= 7 \\
&= 8 \\
&= 2
\end{align*} \]

\(^2\)Crawford and Milenković, 2015
The prior approach has been demonstrated in multiple works\(^3\) using *graphlets*. Our contributions are three-fold:

1. We developed a parallel and optimized alignment algorithm based on this prior work.
2. We investigated its usage with both *graphlets* and *treelets* (to be discussed).
3. We further extended our implementation to also utilize per-edge subgraphs counts based on the recent work of [Crawford and Milenković, 2015].

\(^3\) [Kuchaiev et al., 2010, Milenković et al., 2010, Memisević and Pržulj, 2012, Kuchaiev and Pržulj, 2011, Malod-Dognin and Pržulj, 2015]
Graphlets: All 2-5 undirected induced subgraphs of some larger network. (pictured below)

Treelets: All 3-7 undirected non-induced subgraphs of some larger network.

Figure from [Malod-Dognin and Pržulj, 2015].
Why do we want to use treelets?

There are many benefits to using treelets in lieu of graphlets for this problem;

- **Complexity**: Enumerating graphlets scales with the current fastest algorithm as $O(n \cdot \Delta(G)^4)$, where $n$ is the number of vertices of some graph $G$ and $\Delta(G)$ is the maximum degree. Using efficient algorithms, treelets can be enumerated with low error in about $O(m)$ time, where $m$ is the number of edges of $G$.

- **Scale**: Because of this lower work complexity, tree-structured subgraphs of a larger order relative to graphlets can be enumerated with the same or lower in-practice computational costs. This captures a richer per-vertex feature set for use in alignment.

- **Induced vs. non-induced**: Non-induced subgraph enumeration, as is done with treelets, is much more resilient to the network noise commonly found in real-world biological interaction datasets$^4$.

---

$^4$[Slota and Madduri, 2014]
Numerous parts of the baseline graph alignment algorithms are amenable to parallelization:

- Calculation of pairwise mapping costs \( \forall u, v \in V(H), V(G) \).
- Finding minimum cost vertices \( u, v \) to serve as new seeds for a regional alignment.
- Determining \( k \)-hop neighborhoods of \( u \) and \( v \) for potential alignment pairs.
- Calculating the \( p^{th} \) power of both \( H \) and \( G \).

We perform shared-memory parallelization for all of the above subroutines with OpenMP.
Experimental Setup

- **System**: We run on dual socket Xeon(R) Platinum 8160 CPU node with 196 GB DDR4 and 96 threads.

- **Evaluation**: We evaluate quality and enumeration time for Graphlets, Treelets, and edge-based Treelets.
  - For quality, we use the symmetric substructure score.
  - Basically, the ratio of edges aligned over total edges in both networks minus edges aligned.

- **Networks**: We use protein interaction networks for Yeast, Human, and C.elegans (shown on next slide). For evaluating alignment quality, we noise the Yeast network with 5-20% edge re-wired and align to the original network.
The most promising benefit for future large-scale efforts is the scalability benefit of treelets. We compare against the current state-of-the-art code for counting graphlets (Orca\textsuperscript{5}) and the state-of-the-art for treelets (Fascia\textsuperscript{6}). We observe a considerable scalability difference when counting all subgraphs necessary for alignment computation.

<table>
<thead>
<tr>
<th>Network</th>
<th>n</th>
<th>m</th>
<th>Orca</th>
<th>Fascia</th>
<th>network Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yeast</td>
<td>5.1 K</td>
<td>22 K</td>
<td>4.1s</td>
<td>11s</td>
<td>[Xenarios et al., 2002]</td>
</tr>
<tr>
<td>Human</td>
<td>9.1 K</td>
<td>41 K</td>
<td>9.1s</td>
<td>18s</td>
<td>[Radivojac et al., 2008]</td>
</tr>
<tr>
<td>C.elegans</td>
<td>15 K</td>
<td>246 K</td>
<td>777s</td>
<td>51s</td>
<td>[Cho et al., 2014]</td>
</tr>
</tbody>
</table>

\textsuperscript{5}Hočevar and Demšar [2014]
\textsuperscript{6}Slota and Madduri [2013]
We compare alignment quality using Graphlets, Treelets, and Edge-based Treelet counts (TreeletsEdges) on the noised Yeast networks across various $\alpha$ values. We observe a 3.1% improvement on average using Treelets instead of Graphlets, and a 9.2% improvement when also using edge-based counts.
Conclusions and thanks!

Major takeaways:

- We implement and parallelize prior graph alignment algorithms using *treelet* counts instead of *graphlet* counts.
- We observe a small but measurable increase in alignment quality.
- The more notable benefit is much better scalability to the alignment of larger networks.
- *Future work*: analysis of large-scale biological interaction networks, brain connectome scans, etc. using this code.

Thank you! Contact below with any questions.

slotag@rpi.edu    www.gmslotan.com


