FINAL: Fast Attributed Network Alignment

Presented by Si Zhang (ASU)

Si Zhang
Hanghang Tong
Why Network Alignment?

- Find Someone Like You

- Q: what if someone lives in a different universe (network)?
More Applications

Identify Species-Specific Pathways [Singh’08]

Protein-Protein Interaction (PPI) networks

Cross-Site Recommendation [Zhang’14]

Cross Network Information Diffusion [Zhan’16]

Ontology Matching on Semantic Web [Doan’02]

Arizona State University
Why Network Alignment: How to

- Existing Methods
  - IsoRank [Singh’08], NetAlign [Bayati’09], BigAlign [Koutra’13], UMA [Zhang’15]

- Key Idea: topological consistency
  - Network $G_2$ is a noisy permutation of network $G_1$
  - $G_2 \approx S^T G_1 S$
Topology Consistency: Limitations

- Topological consistency could be easily violated
  - Same nodes may behave differently across different networks
  - Different nodes may have similar connectivity structures

Only topology is not enough!
Topology Consistency: How to Rescue

- Real networks have rich attributes on nodes and/or edges

Q: how to calibrate topology-based alignment by leveraging attributes?
Challenges: Attributed Network Alignment

- C1: Formulation
- C2: Optimality
- C3: Scalable Computation
C1. Formulation

- Typical Network Alignment

<table>
<thead>
<tr>
<th>NetAlign [Bayati’09]</th>
<th>UMA [Zhang’15]</th>
</tr>
</thead>
<tbody>
<tr>
<td>maximize $\alpha h^T s + \left(\frac{\beta}{2}\right) s^T W s$</td>
<td>minimize $|S^T AS - B|_F^2$</td>
</tr>
<tr>
<td>subject to $A s \leq 1$, $s_{ii'} \in {0,1}$</td>
<td>$s.t$ $S1_{n_2 \times 1} \leq 1_{n_1 \times 1}$, $S^T 1_{n_1 \times 1} \leq 1_{n_2 \times 1}$</td>
</tr>
</tbody>
</table>

- **Obs:** only encode topological information

- **Q:** what are their attributed counterparts?
C2. Optimality

- **Obs #1**: many topology-based approaches are non-convex or even NP-hard to find the local minima.

- **Obs #2**: attribute may complicate the optimization problem.

\[
\min_S J(S) = \sum_{a,b,x,y} \left[ \frac{S(x,a)}{\sqrt{f(x,a)}} - \frac{S(y,b)}{\sqrt{f(y,b)}} \right]^2 \times A_1(a,b)A_2(x,y)
\]

without attributes

\[
\min_S J(S) = \sum_{a,b,x,y} \left[ \frac{S(x,a)}{\sqrt{f(x,a)}} - \frac{S(y,b)}{\sqrt{f(y,b)}} \right]^2 A_1(a,b)A_2(x,y) \times \mathbb{1}(N_1(a,a) = N_2(x,x)) \mathbb{1}(N_1(b,b) = N_2(y,y)) \times \mathbb{1}(E_1(a,b) = E_2(x,y))
\]

with attributes

- **Q #1**: what is the exact optimality of the attributed network alignment?

- **Q #2**: how to get the optimal solution, with a comparable complexity (as the topology-alone alignment)?
C3. Scalable Computation

- **Obs #1**: most existing methods have an $O(mn)$ complexity [Singh’08].
- **Obs #2**: best empirical scalability is near-linear [Koutra’13].
- **Q**: how to scale up attributed network alignment?
C3. Scalable Computation

- **Obs**: cross-network search – to find similar users in one network for a given user in another network.

- **Q**: how to speed up on-query network alignment, without solving the full alignment problem?
Outline

▪ Motivations

▪ Q1: FINAL Formulation

▪ Q2: FINAL Algorithms

▪ Q3: FINAL Speed-up Computation

▪ Experimental Results

▪ Conclusions
### Prob. Def: Attributed Network Alignment

- **Given:**
  - (1) two attributed networks $G_1 = \{A_1, N_1, E_1\}$ and $G_2 = \{A_2, N_2, E_2\}$;
  - (2 – optional) a prior alignment preference $H$.

- **Find:** alignment/similarity matrix $S$

### An Illustrative Example

![Illustrative Example Diagram]

**Given** $G_1$ and $G_2$:
- $G_1$ has different shapes as node attributes and straight vs. curved lines as edge attributes.
- $G_2$ has the same node attributes but different edge attributes.

**Find** the alignment/similarity matrix $S$ that shows the alignment between nodes $G_1$ and $G_2$. The matrix $S$ reflects the similarities between nodes, with higher values indicating better alignment.
Prob. Def: On-query Attributed Network Alignment

- **Given:**
  - (1) two attributed networks $G_1 = \{A_1, N_1, E_1\}$ and $G_2 = \{A_2, N_2, E_2\}$;
  - (2 – optional) a prior alignment preference $H$;
  - (3) a query node-$a$ in $G_1$

- **Find:** a vector $s_q$ (similarities of node-$a$ vs. all nodes in $G_2$)
**FINAL Formulation #1: Topological Consistency**

- **Intuition:** similar node-pairs tend to have similar neighboring node-pairs

- **Example:**
  - large $S(a, x)$
  - large $A_1(a, b)$ and $A_2(x, y)$

- **Mathematical Details:** $\min_s [S(a, x) - S(b, y)]^2 A_1(a, x) A_2(b, y)$
**FINAL Formulation #2: Node Attribute Consistency**

- **Intuition:** similar node-pairs share same node attributes

  ![Diagram](image)

- **Example:**
  - large $S(a, x)$ node-$a$ and node-$x$ share same node attribute

- **Mathematical Details:**

  \[
  \min_S [S(a, x) - S(b, y)]^2 A_1(a, x) A_2(b, y)
  \]
**Intuition:** similar node-pairs connect to their neighbor-pairs via same edge attributes

**Example:**
- large $S(a, x)$
- large $S(b, y)$

**Mathematical Details:** if $E_1(a, b) = E_2(x, y)$,

$$\min_S [S(a, x) - S(b, y)]^2 A_1(a, x)A_2(b, y)$$
Putting everything together

- **Objective Function:**

\[
\min_{\mathcal{S}} \quad J(\mathcal{S}) = \sum_{a,b,x,y} \left[ \frac{S(x, a)}{\sqrt{f(x, a)}} - \frac{S(y, b)}{\sqrt{f(y, b)}} \right]^2 \\
\times \mathbb{I}(N_1(a, a) = N_2(x, x)) \mathbb{I}(N_1(b, b) = N_2(y, y)) \\
\times \mathbb{I}(E_1(a, b) = E_2(x, y))
\]

#1. Topology Consistency

\[A_1(a, b)A_2(x, y)\]

#2. Node Attribute Consistency

#3. Edge Attribute Consistency

- **f(x, a):**
  - ‘joint degree’ of node-a and node-x
  - normalization to make the optimization problem convex

- **Generalization:**
  - replacing \(\mathbb{I}(\cdot)\) by an attribute similarity function
  - can handle numerical attributes on nodes and/or edges.
Matrix-Form Objective Function

\[
\min_{s} J(S) = \min_{s} \sum_{v,w} \left[ \frac{s(v)}{\sqrt{D(v, v)}} - \frac{s(w)}{\sqrt{D(w, w)}} \right]^2 W(v, w)
\]

\[
s = \text{vec}(S) = \min_{s} s^T (I - \overline{W}) s
\]

\(- W = N [E \odot (A_1 \otimes A_2)] N \), i.e., the attributed Kronecker product

\(- D \) is the degree matrix of \( W \)

\[
\overline{W} = D^{-\frac{1}{2}} W D^{-\frac{1}{2}} \text{ is the symmetrically normalization of } W
\]
**FINAL Formulation: Matrix Form with Regularization**

- Add a regularization term

\[
\min_s \alpha s^T (I - \hat{W}) s + (1 - \alpha) \| s - h \|^2_2
\]

- \( h \) is default as a uniform vector
- \( h \) encodes the prior knowledge of alignment preferences
- \( h \) avoids trivial solution, e.g., optimal solution \( s = 0 \) w/o \( h \)
- teleport vector in PageRank, restart vector in RWR (on the attributed Kronecker product graph)
Relationship with Existing Methods

- **FINAL vs. IsoRank** [Singh’08]
  - w/o attribute, FINAL = IsoRank (by a scaling factor $D^{1/2}$)

- **FINAL vs. Random Walk Graph Kernel (RWGK)** [Vishwanathan’10]
  - $k(G_1, G_2) = \sum_i q(i)s(i)$, where $q$ is the stopping probability vector

- **FINAL vs. SimRank (Node Proximity)** [Jeh’02]
  - $G_1 = G_2$ and w/o attribute, FINAL = SimRank by a scaling factor $D^{1/2}$

- **FINAL vs. Random Walk with Restart (RWR)** [Tong’06]
  - $s = \text{RWR vector}$ (defined on the attributed Kronecker graph)
Outline

- Motivations
- Q1: FINAL Formulation
- Q2: FINAL Algorithms
- Q3: FINAL Speed-up Computation
- Experimental Results
- Conclusions
**FINAL Solutions**

\[
\min_{s} \alpha s^T (I - \overline{W}) s + (1 - \alpha) \|s - h\|_2^2
\]

- **Obs**: a convex optimization problem
- **Benefits**: a fixed-point solution converging to the global optimal solution
  \[
  s = a \overline{W} s + (1 - \alpha) h \Rightarrow s = (1 - \alpha) (I - \alpha \overline{W})^{-1} h \text{ (closed form)}
  \]
- **Intuition**: a similarity propagation to neighboring node-pairs, which is additionally filtered by node/edge attributes
- **Challenges**: computationally VERY expensive
  - Iterative solution: \(O(m^2 \max_t)\) (due to Kronecker product)
  - Closed form solution: \(O(m^6)\) (due to matrix inversion)
- **Q**: how to scale up and speed up?
Outline

▪ Motivations  ✔
▪ Q1: FINAL Formulation  ✔
▪ Q2: FINAL Algorithms  ✔
▪ Q3: FINAL Speed-up Computation
▪ Experimental Results
▪ Conclusions
FINAL — Speed-up Full Alignment

- **Obs:** FINAL vs. RWGK and RWR
- **Solution:** leverage the existing fast solutions for RWGK and/or RWR [Kang 2012]
- **An Example:** only consider node attributes
  \[
s = (1 - \alpha) \left( I - \alpha D^{-\frac{1}{2}} N (A_1 \otimes A_2) N D^{-\frac{1}{2}} \right)^{-1} h
\]
- **Key Idea:** low rank approximation of $A_1$ and $A_2$
  \[
  A_1 \approx U_1 \Lambda_1 U_1^T, \quad A_2 \approx U_2 \Lambda_2 U_2^T
  \]
  Sherman-Morrison Lemma
  \[
  s \approx (1 - \alpha) \left( I + \alpha D^{-\frac{1}{2}} N U \Lambda U^T N D^{-\frac{1}{2}} \right) h
  \]
  where $U = U_1 \otimes U_2$
  $\Lambda = [(\Lambda_1 \otimes \Lambda_2)^{-1} - \alpha U^T N D^{-1} N U]^{-1}$
- **Complexity:** $O(n^2 r^4)$
- **Challenge:** it is still $O(n^2)$. Can we do better?
FINAL — Speed-up On-query Alignment

**Obs**: only need one column, or one segment of $S$

**Key Ideas**:
- Low-rank approximation (same as for the full alignment)
- Relax the degree matrix $D_N = D_1 \otimes D_2$

**Details**: $s_a = (1 - \alpha) \left[H(:, a) + \alpha(D_1(a, a)D_2)^{-\frac{1}{2}}N_a \right]$ 
\[ \times \left[ (U_1(a, :) \otimes U_2) \hat{\Lambda} U^TN(D_1 \otimes D_2)^{-\frac{1}{2}}h \right] \]

- $(1)$ same trick as for full alignment
- $O(nr^2)$
- $O(n^2r^2)$

\[ g = U^TN (D_1 \otimes D_2)^{-\frac{1}{2}}h = \sum_{i=1}^{p} \sum_{k=1}^{K} \sigma_i \left( U_1^TN_1^k D_1^{-\frac{1}{2}}v_i \right) \otimes \left( U_2^TN_2^k D_2^{-\frac{1}{2}}u_i \right) \]

- $(2)$ SVD on matrix $H = \sum_{i=1}^{p} \sigma_i u_i v_i^T$
- $O(nr)$
- $O(nr)$

**Benefits**: linear complexity $O((Kr^2 + pKr + p^2)n + mr + m_Hp + r^6)$
Outline

- Motivations ✓
- Q1: FINAL Formulation ✓
- Q2: FINAL Algorithms ✓
- Q3: FINAL Speed-up Computation ✓
- Experimental Results
- Conclusions
Experimental Setup

- **Datasets:**
  - DBLP co-author networks (nodes: 9,143 vs. 9,143)
  - Douban online & offline networks (nodes: 3,906 vs. 1,118)
  - Flickr & Last.fm networks (nodes: 12,974 vs. 15,436)
  - Flickr & Myspace networks (nodes: 6,714 vs. 10,733)

- **Evaluation Objectives:**
  - Effectiveness: one-to-one alignment accuracy
  - Efficiency: running time

- **Comparison Methods:**
<table>
<thead>
<tr>
<th>FINAL (Our Methods)</th>
<th>Baseline Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>FINAL-NE (with node &amp; edge attributes)</td>
<td>IsoRank [Singh’08]</td>
</tr>
<tr>
<td>FINAL-N (with node attributes)</td>
<td>NetAlign [Bayati’09]</td>
</tr>
<tr>
<td>FINAL-E (with edge attributes)</td>
<td>UniAlign [Koutra’13]</td>
</tr>
<tr>
<td>FINAL-N+ (speed-up FINAL-N)</td>
<td>Klau’s Algorithm [Klau’09]</td>
</tr>
</tbody>
</table>
R1. Effectiveness Results

**Obs:** attributes help improve network alignment
R2. Quality-Speed Balance

Obs: FINAL gain a better quality-speed balance.
R3. Scalability of FINAL-N+

**Obs:** FINAL-N+ has a quadratic time complexity w.r.t the number of nodes.
**R4. Quality-Speed of FINAL On-Query**

**Obs:** FINAL On-Query gains around 90% accuracy relative to exact FINAL-N, but more than 100 times faster.
R5. Scalability of FINAL On-Query

**Obs:** FINAL On-Query has a **linear** time complexity
Outline

- Motivations
- Q1: FINAL Formulation
- Q2: FINAL Algorithms
- Q3: FINAL Speed-up Computation
- Experimental Results
- Conclusions
Conclusions

- **Attributed Network Alignment**
  - **Q1:** Formulation
  - **A1:** FINAL family
  - **Q2:** Optimality
    - **A2:** Convex optimization problem \( \text{global optimal solution} \)
  - **Q3:** Scalable computation
    - **A3:** Fast algorithms (FINAL-N+ & FINAL On-Query)

- **Results**
  - FINAL outperform other baseline methods
  - FINAL On-Query linear complexity

- **More in Paper**
  - Proof of optimality & more experimental results