Arc-Community Detection via Triangular Random Walks

Paolo Boldi and Marco Rosa
Dipartimento di Informatica
Università degli Studi di Milano

(partly written @ Yahoo! Labs in Barcelona)
Social networks & Communities

• Complex networks exhibit a **finer-grained internal structure**

• Community = **densely connected** set of nodes

• Community detection = partition that optimizes some **quality function**

• **BUT:** rarely a node is part of a **single community**!

• ⇒ **Overlapping communities**
Plan of the talk

• From node-communities to arc-communities?

• Standard vs. Triangular Random Walks

• Using Triangular Random Walks for clustering, through
  • off-the-shelf clustering of the weighted line graph
  • direct implicit clustering (ALP)

• Experiments
Overlapping node clustering vs. arc clustering

• Most algorithms: considering overlapping communities think of overlap as a possibly frequent phenomenon, but stick to the idea that most nodes are well inside a community

• In a large number of scenarios: belonging to more groups is a rule more than an exception

• In a social network, every user has different personas, belonging to different communities...

• ...On the other hand, a friendship relation has usually only one reason!

• ⇒ Arc clustering
Arc-clustering: a metaphorical motivation

Infinitely many lines pass through a single point.
Arc-clustering: a metaphorical motivation

Only one line passes through two points.
Related work - Community detection

- Community detection (possibly with overlaps): too many to mention! [Kernighan & Lin, 1970; Girvan & Newman, 2002; Baumes et al., 2005; Palla et al., 2005; Mishra et al., 2008; Blondel et al., 2008]

- Good surveys / comparisons / analysis: Lancichinetti & Fortunato, 2009; Leskovec et al., 2010; Abrahao et al., 2012

- The latter, in particular, concludes essentially that:
  - different algorithms discover different communities
  - baseline (BFS) performs better than most algorithms (!)
Related work - Link communities


- The latter uses *line graphs* (like we do), but in their undirected version
Random walks (RW) on a graph

• Standard random walk: a sequence of r.v.

\[ X_0, X_1, \ldots \]

such that

\[ P[X_{t+1} = y | X_t = x] = \begin{cases} \frac{1}{d^+(x)} & \text{if } x \rightarrow y \\ 0 & \text{otherwise} \end{cases} \]

• The surfer moves around, choosing every time an arc to follow uniformly at random
Random walks with restart (RWR) on a graph

- Random walk with restart: a sequence of r.v.

\[ X_0, X_1, \ldots \]

such that

\[
P[X_{t+1} = y | X_t = x] = \begin{cases} 
\alpha/d^+(x) + (1 - \alpha)/n & \text{if } x \rightarrow y \\
1 - \alpha/n & \text{otherwise}
\end{cases}
\]

- The surfer every time, with probability \( \alpha \) follows a random arc...

- ...otherwise, teleports to a random location
A graphic explanation of RWR

Surfer at node $x$

Follows a link (to $y$)
uniformly at random

$\alpha$

$1 - \alpha$

Teleports to a random node
Why random walk with restart?

• Teleporting guarantees that there is a unique stationary distribution

• This is not true for standard RW, unless the graph is strongly connected and aperiodic

• Note that the stationary distribution will depend on the damping factor as well

• The stationary distribution of RWR is PageRank
From nodes to arcs

• The stationary distribution of RWR associates a probability $\nu_x$ to every node

• Implicitly, it also associates a probability (frequency) to every arc $x \rightarrow y$:

$$P[X_t = x, X_{t+1} = y] = P[X_{t+1} = y|X_t = x] P[X_t = x] = \nu_x (\alpha / d^+(x) + (1 - \alpha) / n)$$
Triangular random walks (TRW) on a graph

• A TRW is more easily explained dynamically

• A surfer goes from x to y and then to z

• Was there a way to go directly from x to z? If so the move y→z is called triangular step (because it closes a triangle)
A graphic explanation of TRW

Surfer at node x

\[ \alpha \]

Follows a link (to y)
uniformly at random

\[ \beta \]

1 − \beta

Chooses a non-triangular step

Chooses a triangular step

1 − \alpha

Teleports to a random node
TRW: interpretation of the parameters

- $\alpha$ tells you how frequently one follows a link (instead of teleporting)
- $\beta$ tells you how frequently one chooses non-triangles (instead of triangles)

No-teleportation is obtained when $\alpha \rightarrow 1$

There is no choice of $\beta$ that reduces TRW to RWR

One possibility would be to change the definition of a TRW so that $\beta$ is the ratio between the probability of non-triangles and the probability of triangles...

...then one would recover RWR from TRW when $\beta \rightarrow 1$
The idea behind TRW

- Triangular random walks tend to insist differently on triangles than on non-triangles...

- ...you can decide how much more (or less) using $\beta$ as a knob

- The idea is to confine the surfer as long as possible within a community

- Note that when $\beta$ is close to zero, we virtually never choose non-triangular steps...

- ...in such a scenario, the only way out of dense communities is by teleportation
An experiment: Zachary’s Karate Club

\[ \text{TRW, } \beta = 0.01 \]
TRW & Markov chains

- A standard random walk is memoryless: your state at time $t+1$ just depends on your state at time $t$.

- A TRW is a Markov chain of order 2: your state at time $t+1$ depends on your state at time $t$ plus your state at time $t-1$.

- Can we turn it into a standard Markov chain?
Line graphs

- Given a graph $G=(V,E)$, let’s define its (directed) line graph $L(G)=(E,L(E))$ where there is an arc between every node of the form $(x,y)$ and every node of the form $(y,z)$.

- *Theorem:* A TRW on $G$ is a standard RWR on a (weighted version of) $L(G)$.

- Weights depend on the choice of $\beta$.

- Those weights will be denoted by $w_T$.

- “T” is mnemonic for “triangular.”
Second-order weights

- One can compute the stationary distribution (=PageRank) on $L(G)$ using $wT$ as weights...

- This is a distribution on the nodes of $L(G)$ (=arcs of $G$)

- Recall the Karate Club example

- Also induces (as usual) a distribution on its arcs (=pairs of consecutive arcs of $G$)

- This can be seen as another form of weight, denoted by $wS$

- “S” for “Second-order” (or “Stationary”)
Triangular Arc Clustering
(1) Using an off-the-shelf algorithm

• Given G...

• a) compute $L(G)$

• b) weight it (using either $w_T$ or $w_S$)

• c) use any node-clustering algorithm on $L(G)$ that is sensible to weights
Cons and pros of this solution

- **CONs:** The main limit of this solution is **graph size**

- L(G) is larger than G

- If G has \( \approx Ck^{-\gamma} \) nodes of degree \( k \)...

- ...L(G) has \( \approx C^2 k^{-2\gamma} \) nodes of degree \( k \)

- **PROs:** You can use *any* off-the-shelf standard node-clustering algorithm

- Moreover, L(G) turns out to be very easy to compress...

- ...and PageRank converges extremely fast on it
Triangular Arc Clustering
(2) A direct approach (ALP)

- There is no real need to compute \( L(G) \) explicitly!

- One can take a node-clustering algorithm of her will, and have it manipulate \( L(G) \) implicitly

- We did so for \textit{Label Propagation} [Raghavan \textit{et al.}, 2007]
Triangular Arc Clustering

(2) A direct approach (ALP)

- The advantage of LP [Raghavan et al., 2007] with respect to other algorithms is that:
  - it provides a good compromise between quality and speed
  - efficiently parallelizable and suitable for distributed implementations
  - due to its diffusive nature it is very easy to adapt it to run implicitly on the line graph
  - Recently shown that naturally clustered graphs are correctly decomposed by LP [Kothapalli et al., 2012]
Quality measure

• Given a measure $\sigma$ of arc similarity...

• ...and an arc clustering $\lambda$

• The PRI (Probabilistic Rand Index) is

$$PRI(\lambda, \sigma) = \sum_{\lambda(xy) = \lambda(x'y')} \sigma(xy, x'y') - \sum_{\lambda(xy) \neq \lambda(x'y')} \sigma(xy, x'y')$$
Quality measure

- Computing PRI exactly on large graphs is out of question!

- Instead, we sample arcs according to some distribution \( \Psi \)

\[
E_{\Psi} \left[ (-1)^{\lambda(xy) \neq \lambda(x'y')} \sigma(xy) \right]
\]

- If \( \Psi \) is uniform, the value is an unbiased estimator for PRI

- We experiment with: uniform (u), node-uniform (n), node-degree (d)
A) Parameter tuning

- We tuned the parameters $\alpha$ and $\beta$ using different networks
- Consistent results
- We present them on DBLP
  - edge-similarity: TF-IDF of paper titles
A) Parameter tuning

But for small betas, the quality decreases as alpha increases.

Weak dependency on alpha.

Small betas (preference to triangles) always pay off.
B) Quality and computation time

DBLP (6,707,236 arcs)

<table>
<thead>
<tr>
<th>ALP</th>
<th>#clust</th>
<th>PRI u</th>
<th>PRI n</th>
<th>PRI d</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRW</td>
<td>613203</td>
<td>0.74</td>
<td>0.71</td>
<td>0.75</td>
<td>32s</td>
</tr>
<tr>
<td>st. TRW</td>
<td>592562</td>
<td>0.72</td>
<td><strong>0.75</strong></td>
<td><strong>0.75</strong></td>
<td>32s</td>
</tr>
<tr>
<td>RWR</td>
<td>48025</td>
<td>0.02</td>
<td>0.16</td>
<td>0.18</td>
<td>24s</td>
</tr>
<tr>
<td>st. RWR</td>
<td>38498</td>
<td>0.02</td>
<td>0.08</td>
<td>0.03</td>
<td>22s</td>
</tr>
<tr>
<td>-</td>
<td>38498</td>
<td>0.02</td>
<td>0.08</td>
<td>0.03</td>
<td>22s</td>
</tr>
</tbody>
</table>
## B) Quality and computation time

**DBLP (6,707,236 arcs)**

<table>
<thead>
<tr>
<th>Louvain</th>
<th>#clust</th>
<th>PRI u</th>
<th>PRI n</th>
<th>PRI d</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRW</td>
<td>1493</td>
<td>0.01</td>
<td>0.69</td>
<td>0.53</td>
<td>494s</td>
</tr>
<tr>
<td>st. TRW</td>
<td>2116</td>
<td>0.02</td>
<td>0.71</td>
<td>0.53</td>
<td>456s</td>
</tr>
<tr>
<td>RWR</td>
<td>2301</td>
<td>0.01</td>
<td>0.44</td>
<td>0.39</td>
<td>1080s</td>
</tr>
<tr>
<td>st. RWR</td>
<td>232</td>
<td>0.01</td>
<td>0.43</td>
<td>0.39</td>
<td>1028s</td>
</tr>
<tr>
<td>-</td>
<td>250</td>
<td>0.01</td>
<td>0.16</td>
<td>0.15</td>
<td>316s</td>
</tr>
</tbody>
</table>

Suffers of excessive fragmentation.
## B) Quality and computation time

### DBLP (6,707,236 arcs)

<table>
<thead>
<tr>
<th></th>
<th>#clust</th>
<th>PRI u</th>
<th>PRI n</th>
<th>PRI d</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Evans</td>
<td>200</td>
<td>0.01</td>
<td>0.58</td>
<td>0.44</td>
<td>46min</td>
</tr>
<tr>
<td>LINK</td>
<td>1415245</td>
<td>0.28</td>
<td>0.31</td>
<td>0.51</td>
<td>50h</td>
</tr>
<tr>
<td>Infomap</td>
<td>62680</td>
<td>0.05</td>
<td>0.27</td>
<td>0.29</td>
<td>874s</td>
</tr>
<tr>
<td>Louvain (nodes)</td>
<td>6442</td>
<td>0.01</td>
<td>0.28</td>
<td>0.28</td>
<td>13s</td>
</tr>
</tbody>
</table>

Best competitor: LINK (but sloooooow)
B) Quality and computation time

• ALP offers best compromise between quality and computation time

• **Triangular weights** outperform all the others

• **Stationary triangular weights** slightly outperform “normal” ones

• Same behavior on all datasets (not shown here)
Summary

• We introduced a new type of random walk that treats triangles in a preferential way

• We used it to enhance existing community-detection algorithms

• We applied it through off-the-shelf algorithm to the line graph, as well as by implementing an algorithm that never computes the line graph explicitly

• Experiments show that the results obtained have high quality
Future work

• Work out a closed formula for **triangular stationary distribution**

• Apply the triangular weighting to **other problems** (e.g., information spread, influence maximization etc.)

• See if triangular weighting can help explaining better the **structure of social networks**

• See if it is possible to improve existing **models** of social networks
Thanks!

Questions?