Local graph analytics:
beyond characterizing community structure

Michael W. Mahoney

(ICSI, AMP/RISE Lab, and Department of Statistics, UC Berkeley)

Local graph clustering: motivation
Facebook social network: colour denotes class year

Normalized cuts: finds 20% of the graph

Local graph clustering: finds 3% of the graph

Local graph clustering: finds 17% of the graph

Collaboration network

Data: general relativity and quantum cosmology collaboration network, J. Leskovec, J. Kleinberg and C. Faloutsos, ACM TKDD, 1(1), 2007
Global graph clustering: normalized cuts

Data: general relativity and quantum cosmology collaboration network, J. Leskovec, J. Kleinberg and C. Faloutsos, ACM TKDD, 1(1), 2007
Local graph clustering: small clusters

Data: general relativity and quantum cosmology collaboration network, J. Leskovec, J. Kleinberg and C. Faloutsos, ACM TKDD, 1(1), 2007
Current algorithms and running time
Global, weakly and strongly local methods

**Global methods:** $O(\text{volume of graph})$
- The workload depends on the size of the graph

**Weakly local methods:** $O(\text{volume of graph})$
- A seed set of nodes is given
- The solution is locally biased to the input seed set
- The workload depends on the size of the graph

**Strongly local methods:** $O(\text{volume of output cluster})$
- A seed set of nodes is given
- The solution is locally biased to the input seed set
Global, weakly and strongly local methods

Cluster quality

We measure cluster quality using

\[
\text{Conductance} := \frac{\text{number of edges leaving cluster}}{\text{sum of degrees of vertices in cluster}}
\]

- Conductance\((\{A, B\}) = \frac{2}{2 + 2} = \frac{1}{2}\)
- Conductance\((\{A, B, C\}) = \frac{1}{2 + 2 + 3} = \frac{1}{7}\)

- The smaller the conductance value the better
- Minimizing conductance is NP-hard, we use approximation algorithms
Cluster quality

We measure cluster quality using

\[
\text{Conductance} := \frac{\text{number of edges leaving cluster}}{\text{sum of degrees of vertices in cluster}}
\]

- Conductance(\{A,B\}) = \frac{2}{2 + 2} = \frac{1}{2}
- Conductance(\{A,B,C\}) = \frac{1}{2 + 2 + 3} = \frac{1}{7}

- The smaller the conductance value the better
- Minimizing conductance is NP-hard, we use approximation algorithms
Cluster quality

We measure cluster quality using

Conductance := \frac{\text{number of edges leaving cluster}}{\text{sum of degrees of vertices in cluster}}

- The smaller the conductance value the better
- Minimizing conductance is NP-hard, we use approximation algorithms
We measure cluster quality using

\[
\text{Conductance} := \frac{\text{number of edges leaving cluster}}{\text{sum of degrees of vertices in cluster}}
\]

- Conductance(\{A, B\}) = \frac{2}{2 + 2} = \frac{1}{2}
- Conductance(\{A, B, C\}) = \frac{1}{2 + 2 + 3} = \frac{1}{7}

- The smaller the conductance value the better
- Minimizing conductance is NP-hard, we use approximation algorithms
Cluster quality

We measure cluster quality using

\[
\text{Conductance} := \frac{\text{number of edges leaving cluster}}{\text{sum of degrees of vertices in cluster}}
\]

- Conductance({A,B}) = \frac{2}{2 + 2} = \frac{1}{2}
- Conductance({A,B,C}) = \frac{1}{2 + 2 + 3} = \frac{1}{7}

- The smaller the conductance value the better
- Minimizing conductance is NP-hard, we use approximation algorithms
Local graph clustering methods

- MQI (strongly local): Lang and Rao, 2004
- Approximate Page Rank (strongly local): Andersen, Chung, Lang, 2006
- spectral MQI (strongly local): Chung, 2007
- Flow-Improve (weakly local): Andersen and Lang, 2008
- MOV (weakly local): Mahoney, Orecchia, Vishnoi, 2012
- Nibble (strongly local): Spielman and Teng, 2013

- Sweep cut rounding algorithm
Shared memory parallel methods

- **We parallelize 4 strongly local spectral methods + rounding**
  1. Approximate Page Rank ← this talk
  2. Nibble
  3. Deterministic HeatKernel Approximate Page-Rank
  4. Randomized HeatKernel Approximate Page-Rank
  5. Sweep cut rounding algorithm ←← this talk

- **All local methods take various parameters**
  - Parallel method 1: try different parameters independently in parallel
  - Parallel method 2: **parallelize algorithm for individual run**
    - Useful for **interactive** setting where tweaking of parameters is needed
Community structure
Communities in large informatics graphs

Leskovec, Lang, Dasgupta, & Mahoney “Community Structure in Large Networks ...” (2009)
Leskovec, Lang, & Mahoney “Community Structure in Large Networks ...” (2008, 2010)
Mahoney “Algorithmic and Statistical Perspectives on Large-Scale Data Analysis” (2010)

People imagine social networks to look like:

Real social networks actually look like:

How do we know this plot is “correct”?

• (since computing conductance is intractable)
• Lower Bound Result; Structural Result; Modeling Result; Etc.
• Algorithmic Result (ensemble of sets returned by different approximation algorithms are very different)
• Statistical Result (Spectral provides more meaningful communities than flow)

Data are expander-like at large size scales !!!

Size-resolved conductance (degree-weighted expansion) plot looks like:

There do not exist good large clusters in these graphs !!!
NCPs and three types of graphs

Jeub, Balachandran, Porter, Mucha, and Mahoney (2014)

<table>
<thead>
<tr>
<th>Network</th>
<th>Nodes</th>
<th>Edges</th>
<th>$\langle k \rangle$</th>
<th>$\lambda_2$</th>
<th>$\langle C \rangle$</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CA-GrQc</td>
<td>4158</td>
<td>13422</td>
<td>6.5</td>
<td>0.0019</td>
<td>0.56</td>
<td>Coauthorship: arXiv general relativity</td>
</tr>
<tr>
<td>CA-AstroPh</td>
<td>17903</td>
<td>196972</td>
<td>22.0</td>
<td>0.0063</td>
<td>0.63</td>
<td>Coauthorship: arXiv astrophysics</td>
</tr>
<tr>
<td>FB-Johns55</td>
<td>5157</td>
<td>186572</td>
<td>72.4</td>
<td>0.1258</td>
<td>0.27</td>
<td>Johns Hopkins Facebook network</td>
</tr>
<tr>
<td>FB-Harvard1</td>
<td>15086</td>
<td>824595</td>
<td>109.3</td>
<td>0.0094</td>
<td>0.21</td>
<td>Harvard Facebook network</td>
</tr>
<tr>
<td>US-Senate</td>
<td>8974</td>
<td>422335</td>
<td>60.3</td>
<td>0.0013</td>
<td>0.50</td>
<td>Network of voting patterns in U.S. Senate</td>
</tr>
<tr>
<td>US-House</td>
<td>36646</td>
<td>6930858</td>
<td>240.5</td>
<td>0.0002</td>
<td>0.58</td>
<td>Network of voting patterns in U.S. House</td>
</tr>
</tbody>
</table>

Table 1: Six medium-sized networks. For each network, we show the number of nodes and edges in the largest connected component (LCC), the mean degree/strength ($\langle k_i \rangle$), the second-smallest eigenvalue ($\lambda_2$) of the normalized Laplacian matrix, the mean clustering coefficient ($\langle C_i \rangle$), and a description.
NCPs and core-periphery (or not)

Jeub, Balachandran, Porter, Mucha, and Mahoney (2014)
Approximate Page-Rank
Personalized Page-Rank vector

Pick a vertex $u$ of interest and define a vector:
\[ s[u] = 1, \quad s[v] = 0 \quad \forall v \neq u \]
a teleportation parameter $0 \leq \alpha \leq 1$ and $W = AD^{-1}$ then the PPR vector is given by solving:
\[
(1 - \alpha)W + \alpha s e^T)p = p \Leftrightarrow (I - (1 - \alpha)W)p = \alpha s
\]
Approximate Personalized Page-Rank

R. Andersen, F. Chung and K. Lang. Local graph partitioning using Page-Rank, FOCS, 2006

Algorithm idea: iteratively spread probability mass from vector s around the graph.

- r is the residual vector, p is the solution vector
- $\rho > 0$ is tolerance parameter

Run a coordinate descent solver for PPR until: any vertex $u$ satisfies $r[u] \geq -\alpha \rho d[u]$

Initialize: $p = 0$, $r = -\alpha s$

While termination criterion is not met do

1. Choose any vertex $u$ where $r[u] < -\alpha \rho d[u]$

2. $p[u] = p[u] - r[u]$


4. $r[u] = 0$

residual update

Final step: round the solution $p$ using sweep cut.
Approximate Personalized Page-Rank

R. Andersen, F. Chung and K. Lang. Local graph partitioning using Page-Rank, FOCS, 2006

Initialize: \( p = 0, r = -\alpha \)

While termination criterion is not met do

1. Choose any vertex \( u \) where \( r[u] < -\alpha d[u] \)
2. \( p[u] = p[u] - r[u] \)
3. For all neighbours \( v \) of \( u \): \( r[v] = r[v] + (1-\alpha)r[u]A[u,v]/d[u] \)
4. \( r[u] = 0 \)

\[ \frac{\|r\|_1}{\alpha} = 1, \quad \|p\|_1 = 0, \quad \frac{\|r\|_1}{\alpha} + \|p\|_1 = 1 \]
Approximate Personalized Page-Rank

R. Andersen, F. Chung and K. Lang. Local graph partitioning using Page-Rank, FOCS, 2006

\[
\|r\|_1 = 1, \quad \|p\|_1 = 0.1, \quad \|r\|_1 = 1 + \|p\|_1 = 1.1
\]

Initialize: \( p = 0, r = -\alpha \)
While termination criterion is not met do
1. Choose any vertex \( u \) where \( r[u] < -\alpha d[u] \)
2. \( p[u] = p[u] - r[u] \)
3. For all neighbours \( v \) of \( u \): \( r[v] = r[v] + (1-\alpha)r[u]A[u,v]/d[u] \)
4. \( r[u] = 0 \)
Approximate Personalized Page-Rank

R. Andersen, F. Chung and K. Lang. Local graph partitioning using Page-Rank, FOCS, 2006

- Initialize: $p = 0$, $r = -\alpha s$, where $s$ is a probability vector
- While termination criterion is not met do
  1. Choose any vertex $u$ where $r[u] < -\alpha p d[u]$
  2. $p[u] = p[u] - r[u]$
  4. $r[u] = 0$

\[
\frac{\|r\|_1}{\alpha} = 0.9, \quad \|p\|_1 = 0.1, \quad \frac{\|r\|_1}{\alpha} + \|p\|_1 = 1.0
\]
Approximate Personalized Page-Rank

R. Andersen, F. Chung and K. Lang. Local graph partitioning using Page-Rank, FOCS, 2006

\[ \left\| r \right\|_1 \alpha = 0.855, \quad \left\| p \right\|_1 = 0.145, \quad \frac{\left\| r \right\|_1}{\alpha} + \left\| p \right\|_1 = 1.0 \]

Initialize: \( p = 0, r = -\alpha \)
While termination criterion is not met do
1. Choose any vertex \( u \) where \( r[u] < -\alpha p d[u] \)
2. \( p[u] = p[u] - r[u] \)
3. For all neighbours \( v \) of \( u \): \( r[v] = r[v] + (1 - \alpha) r[u] A[u,v] / d[u] \)
4. \( r[u] = 0 \)
Approximate Personalized Page-Rank

R. Andersen, F. Chung and K. Lang. Local graph partitioning using Page-Rank, FOCS, 2006

Initialize: $p = 0, r = -\alpha s$, where $s$ is a probability vector

While termination criterion is not met do

1. Choose any vertex $u$ where $r[u] < -\alpha p d[u]$
2. $p[u] = p[u] - r[u]$
4. $r[u] = 0$

$\|r\|_1 = 0.7897, \quad \|p\|_1 = 0.2103, \quad \frac{\|r\|_1}{\alpha} + \|p\|_1 = 1.0$
Running time APPR

- At each iteration APPR touches a single node and its neighbours
  - Let supp(p) be the support of vector p at termination which satisfies $\text{vol}(\text{supp}(p)) \leq 1/(\alpha \rho)$
  - Overall until termination the work is: $O(1/(\alpha \rho))$ [Andersen, Chung, Lang, FOCS, 2006]

- We store vectors p and r using sparse sets
  - We can only afford to do work proportional to nodes and edges currently touched
  - We used `unordered_map` data structure in STL (Standard Template Library)
  - Guarantees $O(1/(\alpha \rho))$ work
**Variational Perspective**

APPR is an approximation algorithm but what is it minimizing?

\[
\text{minimize } \frac{1 - \alpha}{2} \|Bp\|_2^2 + \alpha \|H(1 - p)\|_2^2 + \alpha \|Zp\|_2^2 + \rho \alpha \|Dp\|_1
\]

where

- B: is the incidence matrix
- Z, H: are diagonal scaling matrices

---


Variational Perspective

• The optimal solution of the l1-reg. problem has local Cheeger-like guarantees.
  ✓ For unweighted graphs this translates to at most 1/\rho non-zeros at optimality.

• The volume of the nodes that are non-zero at optimality is bounded by 1/\rho.
  ✓ The result holds for unweighted graphs as well.

• Proximal gradient descent (standard method in optimization)
  ✓ converges to the solution \textbf{without} touching nodes that are zero at optimality.
  ✓ Running time: O(1/(\rho \alpha) \times \log \text{factor on } \alpha 1/\alpha \text{ and } 1/\rho).


Variational Perspective

• Is accelerated proximal gradient descent a strongly local method?

• If yes, then we expect $O(1/\sqrt{\alpha} \times 1/\rho)$ running time, compared to $O(1/\alpha \rho)$
Shared memory parallelization
Running time: work depth model

Work depth model: J. Jaja. Introduction to parallel algorithms. Addison-Wesley Professional, 1992

Note that our results are not model dependent.

Model
- Work: number of operations required
- Depth: longest chain of sequential dependencies

Let P be the number of cores available.

By Brent’s theorem [1] an algorithm with work W and depth D has overall running time: \( \frac{W}{P} + D \).

In practice \( \frac{W}{P} \) dominates. Thus parallel efficient algorithms require the same work as its sequential version.

Parallel Approximate Personalized Page-Rank

While termination criterion is not met do
1. Choose **ALL (instead of any)** vertex u where \( r[u] < -\alpha \rho \deg[u] \)
2. \( p[u] = p[u] - r[u] \)
3. For all neighbours v of u: \( r[v] = r[v] + (1-\alpha)/(2\deg[u])r[u] \)
4. \( r[u] = (1-\alpha)r[u]/2 \)

- **Asymptotic work remains the same:** \( O(1/(\alpha \rho)) \).
- **Parallel randomized implementation:** work \( O(1/(\alpha \rho)) \) and depth \( O(\log(1/(\alpha \rho))) \).
  - Keep track of two **sparse** copies of p and r
  - Concurrent hash table for sparse sets \( \leftarrow \) important for \( O(1/(\alpha \rho)) \) work
  - Use atomic increment to deal with conflicts
  - Use of Ligra (Shun and Blelloch 2013) to process only “active” vertices and their edges
- **Same theoretical graph clustering guarantees**, Fountoulakis et al. 2016.
## Data

<table>
<thead>
<tr>
<th>Input graph</th>
<th>Num. vertices</th>
<th>Num. edges</th>
</tr>
</thead>
<tbody>
<tr>
<td>soc-JL</td>
<td>4,847,571</td>
<td>42,851,237</td>
</tr>
<tr>
<td>cit-Patents</td>
<td>6,009,555</td>
<td>16,518,947</td>
</tr>
<tr>
<td>com-LJ</td>
<td>4,036,538</td>
<td>34,681,189</td>
</tr>
<tr>
<td>com-Orkut</td>
<td>3,072,627</td>
<td>117,185,083</td>
</tr>
<tr>
<td>Twitter</td>
<td>41,652,231</td>
<td>1,202,513,046</td>
</tr>
<tr>
<td>Friendster</td>
<td>124,836,180</td>
<td>1,806,607,135</td>
</tr>
<tr>
<td>Yahoo</td>
<td>1,413,511,391</td>
<td>6,434,561,035</td>
</tr>
</tbody>
</table>
Performance

- Slightly more work for the parallel version
- Number of iterations is significantly less
Performance

- 3-16x speed up
- Speedup is limited by small active set in some iterations and memory effects
Network community profile plots

Friendster, 124M nodes, 1.8B edges
Yahoo, 1.4B nodes, 6.4B edges

\[ \mathcal{O}(10^5) \] approximate PPR problems were solved in parallel for each plot,

Agrees with conclusions of [Leskovec et al. 2008], i.e., good clusters tend to be small.
Rounding: sweep cut

- Round returned vector $p$ of approximate PPR
  - **1st step ($O(1/(\alpha \rho) \log(1/(\alpha \rho)))$ work):** Sort vertices by non-increasing value of non-zero $p[u]/d[u]$
  - **2nd step ($O(1/(\alpha \rho))$ work):** Look at all prefixes of sorted order and return the cluster with minimum conductance,

```
Sorted vertices: \{A,B,C,D\}
```

```
<table>
<thead>
<tr>
<th>Cluster</th>
<th>Conductance</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A}</td>
<td>1</td>
</tr>
<tr>
<td>{A,B}</td>
<td>1/2</td>
</tr>
<tr>
<td>{A,B,C}</td>
<td>1/7</td>
</tr>
<tr>
<td>{A,B,C,D}</td>
<td>3/11</td>
</tr>
</tbody>
</table>
```
Parallel sweep cut

• **1st step:** Sort vertices by non-increasing value of non-zero $p[u]/d[u]$.
  - Use parallel sorting algorithm, $O(1/(\alpha \rho) \log(1/(\alpha \rho)))$ work and $O(\log(1/(\alpha \rho)))$ depth.

• **2nd step:** Look at all prefixes of sorted order and return the cluster with minimum conductance.
  - Naive implementation: for each sorted prefix compute conductance, $O((1/(\alpha \rho))^2)$.
  - **We design a parallel algorithm based on integer sorting and prefix sums that takes** $O(1/(\alpha \rho))$ time.
  - **The algorithm computes the conductance of ALL sets with a single pass over the nodes and the edges.**
Parallel sweep cut: 2nd step

Incidence matrix B

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-B</td>
<td>1</td>
<td></td>
<td>-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-C</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-1</td>
</tr>
<tr>
<td>B-C</td>
<td></td>
<td>1</td>
<td></td>
<td>-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C-D</td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-1</td>
</tr>
<tr>
<td>D-E</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td>-1</td>
</tr>
<tr>
<td>D-F</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td>-1</td>
</tr>
<tr>
<td>D-G</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>F-H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

Sorted vertices: \{A, B, C, D\}

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Sum cols B</th>
<th>Volume</th>
<th>Conductance</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A}</td>
<td>2</td>
<td>2</td>
<td>2/2=1</td>
</tr>
<tr>
<td>{A, B}</td>
<td>2</td>
<td>4</td>
<td>2/4=1/2</td>
</tr>
<tr>
<td>{A, B, C}</td>
<td>1</td>
<td>7</td>
<td>1/7</td>
</tr>
<tr>
<td>{A, B, C, D}</td>
<td>3</td>
<td>11</td>
<td>3/11</td>
</tr>
</tbody>
</table>

- Sort vertices
  - work: $O(1/(\alpha \rho) \log(1/(\alpha \rho)))$, depth: $O(\log(1/(\alpha \rho)))$
- Represent matrix B with a sparse set using vertex identifiers and the order of vertices
  - work: $O(1/(\alpha \rho))$, depth: $O(\log(1/(\alpha \rho)))$
- Use prefix sums to sum elements of the columns
  - work: $O(1/(\alpha \rho))$, depth: $O(\log(1/(\alpha \rho)))$
Parallel sweep cut: performance

![Graph showing running time vs number of cores for parallel and sequential sweeps.]
Node Embeddings
Locality and Structure Aware Graph Node Embedding (Lasagne)

• Embed nodes into lower dimensional spaces in an unsupervised way
  ✓ Takes into account local structure

• Useful for
  ✓ Multi-label classification (experiments follow in next slides)

• What is the method?
  ✓ Run local graph clustering from each node (runs in nearly linear time)
  ✓ Get context for each node by sampling neighbors using the Personalized PageRank vector of each node.
  ✓ Build a context matrix for word-to-vec model.
  ✓ Train the word-to-vec model.
Datasets

• Protein-Protein Interactions (PPI):
  This is a subgraph of the PPI network for Homo Sapiens.

• BlogCatalog:
  This is a social network graph where each of the 10,312 nodes corresponds to a user and the 333,983 edges represent the friendship relationships between bloggers. 39 different interest groups provide the labels.

• IMDb Germany:
  This kind of artificial dataset is created from the IMDb movie database. It consists of 32,732 nodes, 1,175,364 edges and 27 labels. Each node represents an actor/actress who played in a german movie. Edges connect actors/actresses that were in a cast together and the node labels represent the genres that the corresponding actor/actress played.

• Flickr:
  The Flickr network is a quite dense social network graph with 80,513 nodes and 5,899,882 edges. Each node describes a user and the links represent friendships.
Datasets Overview

Blogcatalog
- Social Network
- Nodes: 10312 (Blogger)
- Edges: 333983 (Friendship links)
- Considered classes: 29 (Blog categories)

PPI
- Protein-Protein-Interaction Network
- Nodes: 3890 (Proteins)
- Edges: 38739 (Interactions)
- Considered classes: 34 (Biological states)
Datasets Overview

IMDb Germany
- Collaboration Network
- Nodes: 32732 (Actors)
- Edges: 1175364 (Collaborations)
- Considered classes: 25 (Genres of the movies)

Flickr
- Social Network
- Nodes: 80513 (Users)
- Edges: 5899882 (Friendship links)
- Considered classes: 195 (Interest group memberships)
Word2Vec [Mikolov, 2013]

- Learning word representations technique from NLP
- Word representations are learned based on their context (Distributional Hypothesis - words in similar contexts are similar)
  
  ... how to stop puppy from barking...
  ... barking dog stole my sleep...

$\mathbf{x}^T$

\[
\log \sigma(v'_c^T \cdot v_{w_E}) + \sum_{i=1}^{k} E_{w_i \sim P_n(w)}[\log \sigma(-v'_c^T \cdot v_{w_E})]
\]

https://www.tensorflow.org/versions/r0.12/tutorials/word2vec/index.html
Recent “node embedding” work

DeepWalk [Perozzi, 2014]
- Adaptation of word2vec to graphs
- Learning representations of nodes in the graph
- ‘Sentences’ are represented by random walks: $n$ random walks of size $t$
  - The sequence of nodes in random walk is interpreted as “sentence”
  - For each node in the random walk $w$ nodes visited previously to it and $w$ nodes visited after it are interpreted as its context
- Evaluation with Multi-Label classification

Line [Tang, 2015]
- Separately earns representations based on direct neighbours and neighbours of direct neighbours
- Final representation vector is the concatenation of both representation

node2vec [Grover, 2016]
- Different sampling strategy for random walk (Breadth vs. Depth first)
- DFS learns Homophily (highly interconnected nodes are similar)
- BFS learns Structural Equivalence (nodes with similar structural roles are similar)
- Combination of both for the random walk (2nd order random walk):
Recent “node embedding” work

- The main difference is how the neighborhood is explored

- Strong assumptions about neighborhood structure:
  - Distance to the relevant neighbours
  - Neighborhoods of all nodes follow the same pattern

- node2vec:
  - Additional parameters p and q
  - Preprocessing quadratic in node degree
Lasagne

- Node embedding based on Personal PageRank (PPR) with the node as only seed (adaptation of ACL06 algorithm)

- PPR describes the local neighborhood
- Only assumption is the level of locality: teleportation parameter
- Sampling of training instances using PPR entries as weights

- Captures locality more accurately

- Instead of skip-gram/cbow:
  - Node as part of own context
  - Discard own weight, replace through the second largest
NCP plots of datasets

Figure 5: NCP plots for used datasets. Red, solid lines sketch the community structure of the original graph. Blue, dashed lines plot the structure of randomly rewired networks.
Multi-label classification

Figure 6: $F_1$ macro scores for BlogCatalog

Figure 7: $F_1$ macro scores for PPI

Figure 8: $F_1$ macro scores for IMDb Germany

Figure 9: $F_1$ macro scores for Flickr
Social Models
Decision Making

- Predicting choices of individuals is in high demand for computational social sciences, economics etc.
- Digital networking facilitates information flow and spread of influence among individuals.
- **Objective: social graph regularization for latent class discrete choice models.**
  - Individuals with an edge in the social network have higher probability of having the same latent class.

Graphical model (plate notation)

Combines

- Expressiveness of parametric modeling
- Descriptive exploratory power of latent class
- Social network regularization

Extends

- the range of inferences possible with the state-of-the-art discrete choice models
Preliminary Results

(a) Start of the study (Feb 1995): 127 non-smokers (blue), 23 smokers (red), 10 unobserved (yellow) 422 edges.
(b) End of the study (Jan 1997): 98 non-smokers (blue), 39 smokers (red), 23 unobserved (yellow) 339 edges.

Figure 5: Social graphs of student friendships and smoking behaviors within the 2 years period of the study.

<table>
<thead>
<tr>
<th>Summary</th>
</tr>
</thead>
</table>
- Start of study: students are not influenced by smoker friends, prediction is similar for all models.
- End of study: some students were influenced by smoker friends, prediction is better for the social latent class model.

| Table 3: Adolescent smoking prediction, February 1995 |
|-----------------|-----------------|
| model           | accuracy        |
| logistic regression | 81.1%          |
| latent class logistic regression | 78.9%          |
| social logistic regression | 80.0%          |
| social latent class logistic regression | 82.2%          |

| Table 4: Adolescent smoking prediction, January 1997 |
|-----------------|-----------------|
| model           | accuracy        |
| logistic regression | 68.5%          |
| latent class logistic regression | 72.1%          |
| social logistic regression | 65.5%          |
| social latent class logistic regression | 77.1%          |

Reference
Social Mobility (ongoing work)

- Activity based travel demand models are essential tools used in transportation planning and regional development scenario evaluation.

- Activity prediction is performed by using cell phone data, i.e., call detail records and GPS data.

- **Objective: incorporate social influence in activity prediction tasks.**
  - Construct social graphs using cell phone data which together with GPS data are feed into a Long Short-Term Memory neural network for activity prediction.
Real-time Personalized Prediction (ongoing work)

- **Objective:** develop models with personalized solution, i.e., personalized solutions correspond to higher likelihood.
- **Real-time:** training the model requires “local” running time since only highly influential individuals are touched.
- Does not require clipping the graph a-priori, we let the optimal solution to “decide” which individuals are the most important.
- Data and social graph are considered within a single model, i.e., not a two-stage procedure.

**Personalize the solution by localizing the social graph**

**New solvers with local running time**

- By personalizing starting point of the algorithm
- Maintaining local operations per iteration
- Early termination, i.e., free lunch

**Preliminary work**

Conclusions

• Local spectral methods
  ✓ Variants of usual global spectral methods that are biased toward a small part of large data.
  ✓ Strong algorithmic and statistical theory.
  ✓ Very good in practice, e.g., characterizing community structure.

• Beyond community structure.
  ✓ Variational perspective: unifying framework and several improved variants.
  ✓ Shared memory parallel implementations of billion node graphs.
  ✓ Combine with NLP w2v ideas for better node embedding and classification.
  ✓ Starting to combine with social discrete choice and real-time prediction models