A Framework for the Evaluation and Management of Network Centrality

Vatche Ishakian, Dora Erdos, Evimaria Terzi, Azer Bestavros

Department of Computer Science

Boston University
What is the busiest stop?
What is the busiest stop?
Where to add new flights so that the number of travellers is maximized?
Outline

Motivation

General framework for computing centrality

Centrality of nodes

Centrality of Groups

Graph modifications to change centrality values

Experiments
Outline

Motivation

General framework for computing centrality

Centrality of nodes

Centrality of Groups

Graph modifications to change centrality values

Experiments
Outline

Motivation

General framework for computing centrality

Centrality of nodes

Centrality of Groups

Graph modifications to change centrality values

Experiments
Centrality

- \( G(V,E) \) directed acyclic graph
- \( S \subseteq V \) set of source nodes
- \( T \subseteq V \) set of target nodes
- \( P \) set of special paths connecting nodes in \( S \) with \( T \)

\( P_v(s,t) \): set of special paths between source \( s \) and target \( t \) covered by node \( v \).

Centrality of node \( v \) is a function of the paths in \( P \) that \( v \) covers:

\[
C(v) = \sum_{s,t} F(P_v(s,t))
\]
Centrality – Stress centrality - NumPaths

\[ C(v) = \sum_{s,t} \sum_{s,v} F(Pr_v(s, t)) = \sum_{s,t} \left| P_v(s, t) \right| \]

Compute \( C(B) \)
Centrality – Stress centrality - NumPaths

\[ C(v) = \sum_{s,v} |P_v(s,t)| \]

Compute C(B)
Centrality – Stress centrality - NumPaths

\[ C(v) = \sum_{s,v} |P_v(s, t)| \]

Compute \( C(B) \)
Centrality – Stress centrality - NumPaths

\[ C(v) = \sum_{s,v} |P_v(s, t)| \]

Compute C(B)
Centrality – Stress centrality - NumPaths

\[ C(v) = \sum_{s,v} |P_v(s,t)| \]

Compute C(B)
Centrality – Stress centrality - NumPaths

\[ C(v) = \sum_{s,v} | P_v(s,t) | \]

Compute \( C(B) = 8 \)
Centrality – Stress centrality - NumPaths

\[ C(v) = \sum_{s,v} | P_v(s, t) | \]

Compute \( C(B) = 8 \)

Compute \( C(B) \) in a smarter way
Centrality – Stress centrality - NumPaths

\[ C(v) = \sum_{s,v} |P_{v}(s,t)| \]

Compute \( C(B) = 8 \)

Compute \( C(B) \) in a smarter way

Prefix(B) = \# of paths from S to B = 4
Centrality – Stress centrality - #P

\[ C(v) = \sum_{s,v} \left| P_v(s,t) \right| \]

Compute \( C(B) = 8 \)

Compute \( C(B) \) in a smarter way

Prefix(B) = # of paths from S to B = 4

Suffix(B) = # of paths from B to T = 2

\[ C(B) = \text{Prefix}(B) \times \text{Suffix}(B) \]
Centrality – Stress centrality - NumPaths

\[ C (v) = \sum_{s, v} | P_v (s, t) | \]

C(v) = Prefix(v) x Suffix(v)

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Suffix</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>B</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>C</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>D</td>
<td>8</td>
<td>1</td>
</tr>
<tr>
<td>E</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>F</td>
<td>4</td>
<td>1</td>
</tr>
</tbody>
</table>
Centrality – Betweenness centrality - NumShortestPaths

\[
C(v) = \sum_{s,v} | P_v(s, t) |
\]

Compute \( C(B) \)

[Brandes 2001, 2008]
Centrality – Betweenness centrality - NumShortestPaths

\[ C(v) = \sum_{s,v} |P_v(s, t)| \]

Compute \( C(B) \)

[Brandes 2001, 2008]
Centrality – Betweenness centrality - NumShortestPaths

\[ C(v) = \sum_{s,v} |P_{v}(s,t)| \]

Compute \( C(B) \)

[Brandes 2001, 2008]
Centrality – Betweenness centrality - NumShortestPaths

\[ C(v) = \sum_{s,v} |P_v(s, t)| \]

Compute C(B)

[Brandes 2001, 2008]
Centrality – Betweenness centrality - NumShortestPaths

\[ C(v) = \sum_{s,v} |P_v(s, t)| \]

Compute \( C(B) = 4 \)

[Brandes 2001, 2008]
Centrality – Betweenness centrality - NumShortestPaths

\[ C(v) = \sum_{s,v} |P_v(s,t)| \]

\[ C(v) = \text{Prefix}(v) \times \text{Suffix}(v)^T \]

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Suffix</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>(1,1,1)</td>
<td>(0,4,0)</td>
</tr>
<tr>
<td>B</td>
<td>(1,1,0)</td>
<td>(2,2,0)</td>
</tr>
<tr>
<td>C</td>
<td>(0,1,1)</td>
<td>(0,2,2)</td>
</tr>
<tr>
<td>D</td>
<td>(1,2,1)</td>
<td>(1,1,1)</td>
</tr>
<tr>
<td>E</td>
<td>(1,1,0)</td>
<td>(2,2,0)</td>
</tr>
<tr>
<td>F</td>
<td>(0,1,1)</td>
<td>(0,2,2)</td>
</tr>
</tbody>
</table>
Centrality – Betweenness centrality - NumShortestPaths

\[ C(v) = \sum_{s,v} | P_v(s,t) | \]

\[ C(v) = \text{Prefix}(v) \times \text{Suffix}(v)^T \]

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Suffix</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>(1,1,1)</td>
<td>(0,4,0)</td>
</tr>
<tr>
<td>B</td>
<td>(1,1,0)</td>
<td>(2,2,0)</td>
</tr>
<tr>
<td>C</td>
<td>(0,1,1)</td>
<td>(0,2,2)</td>
</tr>
<tr>
<td>D</td>
<td>(1,2,1)</td>
<td>(1,1,1)</td>
</tr>
<tr>
<td>E</td>
<td>(1,1,0)</td>
<td>(2,2,0)</td>
</tr>
<tr>
<td>F</td>
<td>(0,1,1)</td>
<td>(0,2,2)</td>
</tr>
</tbody>
</table>
Centrality – Betweenness centrality - NumShortestPaths

\[ C(v) = \sum_{s,v} | P_v(s,t) | \]

\[ C(v) = \text{Prefix}(v) \times \text{Suffix}(v)^T \]

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Suffix</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>(1,1,1)</td>
<td>(0,4,0)</td>
</tr>
<tr>
<td>B</td>
<td>(1,1,0)</td>
<td>(2,2,0)</td>
</tr>
<tr>
<td>C</td>
<td>(0,1,1)</td>
<td>(0,2,2)</td>
</tr>
<tr>
<td>D</td>
<td>(1,2,1)</td>
<td>(1,1,1)</td>
</tr>
<tr>
<td>E</td>
<td>(1,1,0)</td>
<td>(2,2,0)</td>
</tr>
<tr>
<td>F</td>
<td>(0,1,1)</td>
<td>(0,2,2)</td>
</tr>
</tbody>
</table>
Centrality – Paths

\[ C(v) = \sum_{s, v} \delta \left( | P_v(s, t) | > 0 \right) \]
Centrality – Paths

\[ C(v) = \sum_{s,v} \delta(|P_v(s, t)| > 0) \]

\[ C(v) = \text{Prefix}(v) \times \text{Suffix}(v)^T \]

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Suffix</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>(1,1,1)</td>
<td>(1,1,1)</td>
</tr>
<tr>
<td>B</td>
<td>(1,1,1)</td>
<td>(1,1,1)</td>
</tr>
<tr>
<td>C</td>
<td>(1,1,1)</td>
<td>(1,1,1)</td>
</tr>
<tr>
<td>D</td>
<td>(1,1,1)</td>
<td>(1,1,1)</td>
</tr>
<tr>
<td>E</td>
<td>(1,1,1)</td>
<td>(1,1,1)</td>
</tr>
<tr>
<td>F</td>
<td>(1,1,1)</td>
<td>(1,1,1)</td>
</tr>
</tbody>
</table>
Centrality – ShortestPaths

\[ C(v) = \sum_{s,v} \delta (|P_v(s,t)| > 0) \]
Centrality – ShortestPaths

\[ C(v) = \sum_{s,v} \delta \left( |P_v(s,t)| > 0 \right) \]

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Suffix</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>(1,1,1)</td>
<td>(0,1,0)</td>
</tr>
<tr>
<td>B</td>
<td>(1,1,0)</td>
<td>(1,1,0)</td>
</tr>
<tr>
<td>C</td>
<td>(0,1,1)</td>
<td>(0,1,1)</td>
</tr>
<tr>
<td>D</td>
<td>(1,1,1)</td>
<td>(1,1,1)</td>
</tr>
<tr>
<td>E</td>
<td>(1,1,0)</td>
<td>(1,1,0)</td>
</tr>
<tr>
<td>F</td>
<td>(0,1,1)</td>
<td>(0,1,1)</td>
</tr>
</tbody>
</table>

\[ C(v) = \text{Prefix}(v) \times \text{Suffix}(v)^T \]
Computing node centrality

Prefix(v) = # of paths from nodes in S to v
Suffix(v) = # of paths from v to nodes in T

C(v) = Prefix(v) x Suffix(v)^T
Computing node centrality – Computing Prefix( )

Prefix(v) = # of paths from nodes in S to v

Prefix(v) can be computed as the sum of the Prefix in v’s parents $\Pi_v$:

$$\text{Prefix}(v) = \sum_{u \in \Pi_v} \text{Prefix}(u)$$
Computing node centrality – Computing Prefix( )

Prefix(v) = # of paths from nodes in S to v

Prefix(v) can be computed as the sum of the Prefix in v’s parents Πv:

\[
\text{Prefix}(v) = \sum_{u \in \Pi_v} \text{Prefix}(u)
\]

1. Fix topological order σ of nodes.

σ = (S1, S2, S3, A, B, C, D, E, F, T)

2. Traverse nodes in order of σ to compute Prefix
Computing node centrality – Computing Prefix( )

Prefix(v) = # of paths from nodes in S to v

Prefix(v) can be computed as the sum of the Prefix in v’s parents $\Pi_v$:

$$\text{Prefix}(v) = \sum_{u \in \Pi_v} \text{Prefix}(u)$$

1. Fix topological order $\sigma$ of nodes.

$\sigma = (S1, S2, S3, A, B, C, D, E, F, T)$

2. Traverse nodes in order of $\sigma$ to compute Prefix
Computing node centrality

\[ C(v) = \text{Prefix}(v) \times \text{Suffix}(v)^T \]

Small changes in the update of \text{Prefix}() and \text{Suffix}() tailor this general computation to the different instances of centrality.

\textbf{NumShortestPaths} : Only consider parents that are on a shortest path
\textbf{Paths} : Use boolean addition/union
\textbf{ShortestPaths} : Only consider parents on shortest paths and use boolean addition/union
Outline

Motivation

General framework for computing centrality

Centrality of nodes

Centrality of Groups

Graph modifications to change centrality values

Experiments
Group centrality

Q: Which two nodes have the largest total NumPaths centrality?

Nodes A and B (or C or D) have the highest NumPaths-centrality

\[ C(A, B) = 14 \]

[Everett, Borgatti, ‘99]
Group centrality

Q: Which two nodes have the largest total NumPaths centrality?

Nodes A and B (or C or D) have the highest NumPaths-centrality

\[ C(A,B) = 14 \]

Nodes B and C can cover all paths

[Everett, Borgatti, ‘99]
Group centrality

Q: Which two nodes have the largest total NumPaths centrality?

Nodes A and B (or C or D) have the highest NumPaths-centrality

C (A,B) = 14

Nodes B and C can cover all paths

C(B,C) = 16

Set{B,C} has the highest group centrality.

[Everett, Borgatti, ’99]
**Group centrality**

U = \{ u_1, u_2, ..., u_k \} set of k nodes.

\( P_U(s, t) \): set of special paths between source s and target t covered by some node in U.

**Group Centrality** of set U is a function of the paths in P that any node in U covers:

\[
C(U) = \sum_{s,t} F(P_U(s,t))
\]
Group centrality – k-Group Centrality Maximization problem (k-GCM)

Optimization problem:

Given graph G(V,E) and integer k find the set of k nodes with highest group centrality.

- k-GCM is NP-hard for NumShortestPaths\(^1\), NumPaths, ShortestPaths centrality
- Objective function is monotone submodular for these centralities
- **Greedy**-type heuristic yields (1-1/e)-approximation algorithm

\(^1\)Dolev et al. 2009
Outline

Motivation

General framework for computing centrality

Centrality of nodes

Centrality of Groups

Graph modifications to change centrality values

Experiments
Outline

Motivation

General framework for computing centrality

  Centrality of nodes

  Centrality of Groups

Graph modifications to change centrality values

Experiments
Where to add new flights so that the number of travellers is maximized?
Outline

Motivation

General framework for computing centrality

Centrality of nodes

Centrality of Groups

Graph modifications to change centrality values

Experiments
Experiments

MemeTracker\textsuperscript{1} dataset – a network of online media sites, where edges correspond to hyperlinks. We choose a directed acyclic subgraph with 20K nodes and 80K edges.

\textsuperscript{1}[Leskovec et al. 2009]
Experiments – baseline algorithms

**Greedy_Sampled**: take a sample of the graph by removing edges at random. Then apply our Greedy algorithm to the sampled graph

**Greedy_max**: pick $k$ nodes with largest individual centrality values

**Greedy_1**: pick $k$ nodes with highest $d_{in} \times d_{out}$

**Random_k**: pick $k$ nodes uniformly at random
Experiments – NumShortestPaths - MemeTracker

Coverage Ratio – performance of baseline algorithm compared to our greedy algorithm

\[ CR = \frac{C(A_{Baseline})}{C(A_{Greedy})} \]
References

Thank You!