Similarity Estimation in Social Networks

Edith Cohen[1] Daniel Delling[1] Fabian Fuchs[2]

Andrew V. Goldberg[1] Moises Goldszmidt[1] Renato Werneck[1]

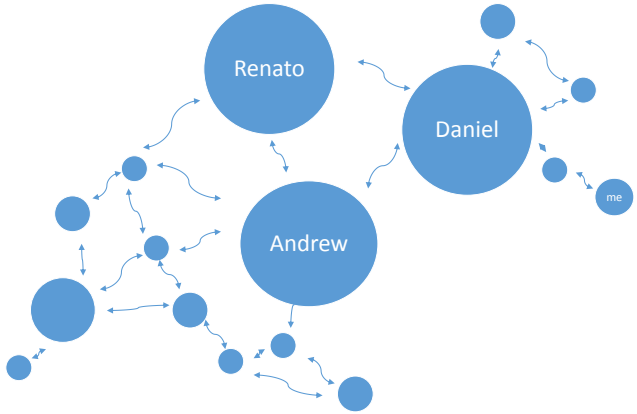
[1]Microsoft Research Silicon Valley

[2]Karlsruhe Institute of Technology, Germany,
intern at Microsoft Research during this work

COSN'13

10/8/2013

Applications in Social Graphs



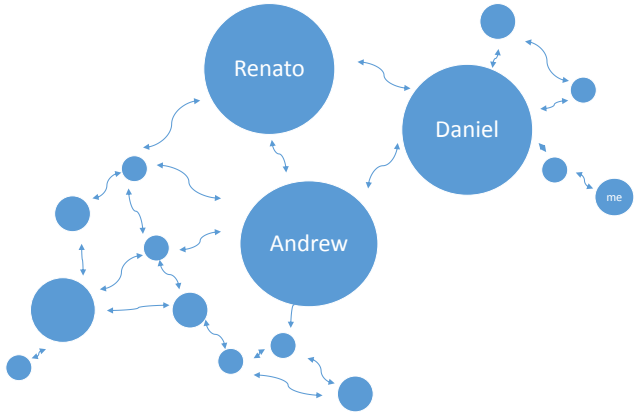
(Some) applications:

- link prediction
- (overlapping) community detection
- seeded communities
- evolution of friendship/similarity
- ...

Important building block

- measure similarity of two nodes

Similarity Measures



Common structural approaches:

- local: Considering all neighbors
  - fast
  - how to handle global queries?
  - Examples: Neighborhood intersection, Adamic-Adar, ...
- global: Considering the whole network
  - more accurate
  - slow on large networks
  - Examples: Random walk with restarts (rooted PageRank), Katz, ...

Remarks:

- we use metadata solely as ground truth for evaluation
- could be combined

Our Goal

A similarity measure:

- as accurate as global measures and as fast as local measures

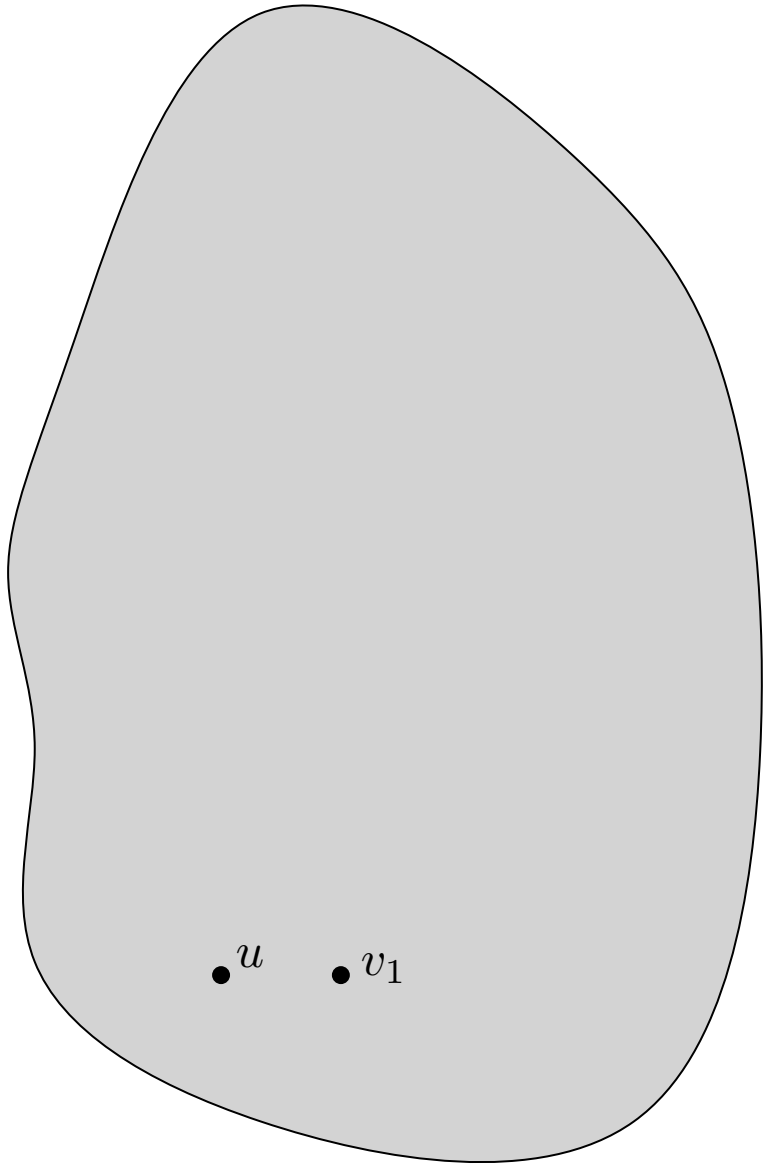
Three ideas:

- evaluate how each of the two nodes views the full graph
  $\rightarrow$ accuracy
- sketch these views
  $\rightarrow$ efficiency
- extend measure to properly deal with multiplicity of links
  $\rightarrow$ resilience

Idea 1: Evaluating the View

Idea:

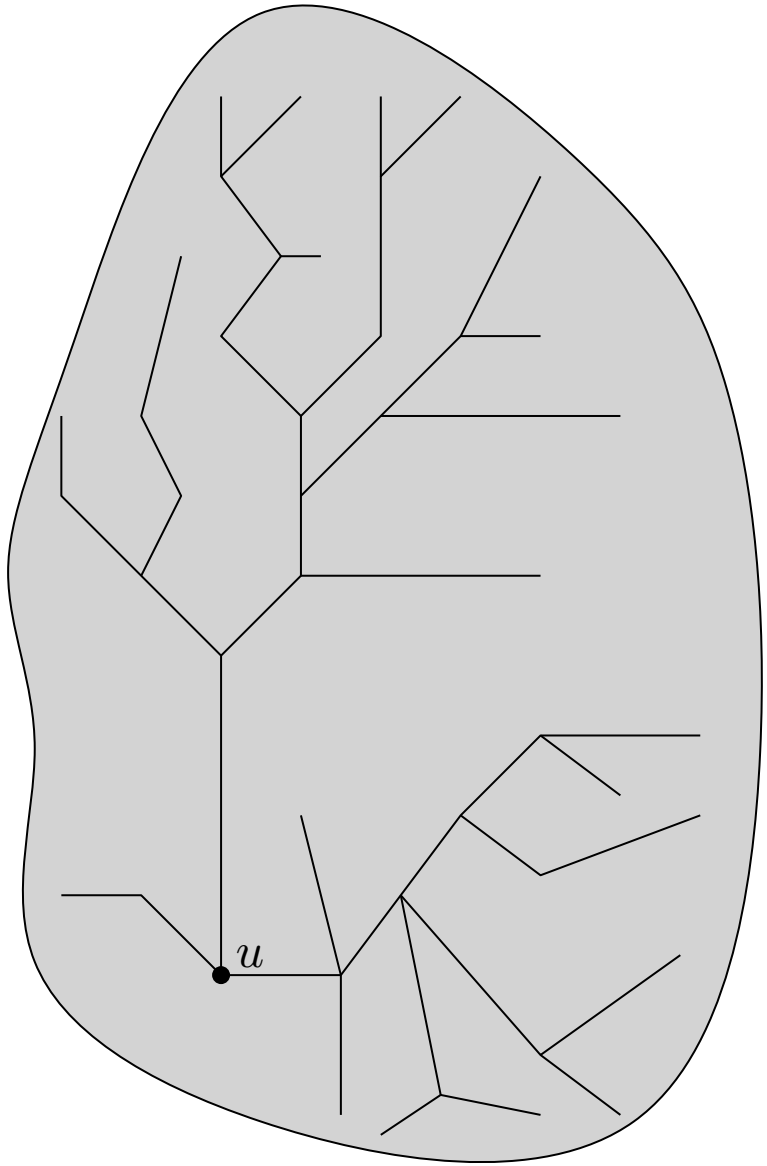
- for nodes u and v
- build shortest path trees from u and v
- compare them
- if they look similar, u and v are similar



Idea 1: Evaluating the View

Idea:

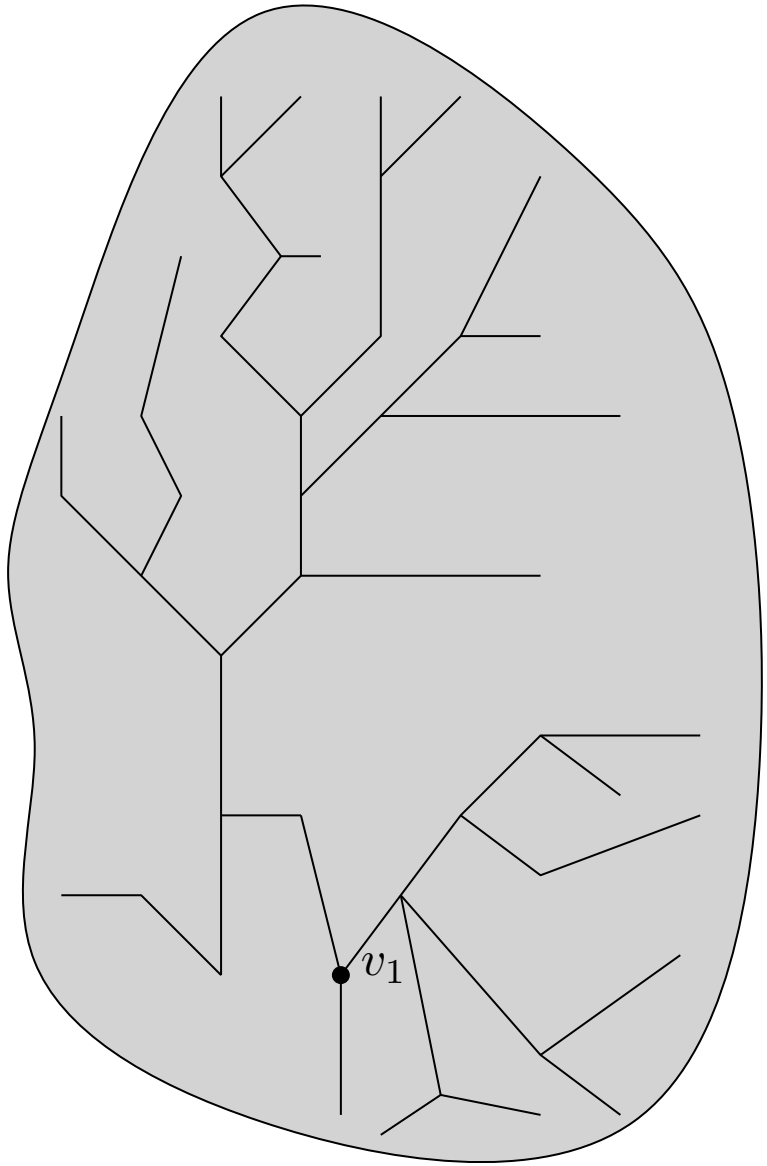
- for nodes u and v
- build shortest path trees from u and v
- compare them
- if they look similar, u and v are similar



Idea 1: Evaluating the View

Idea:

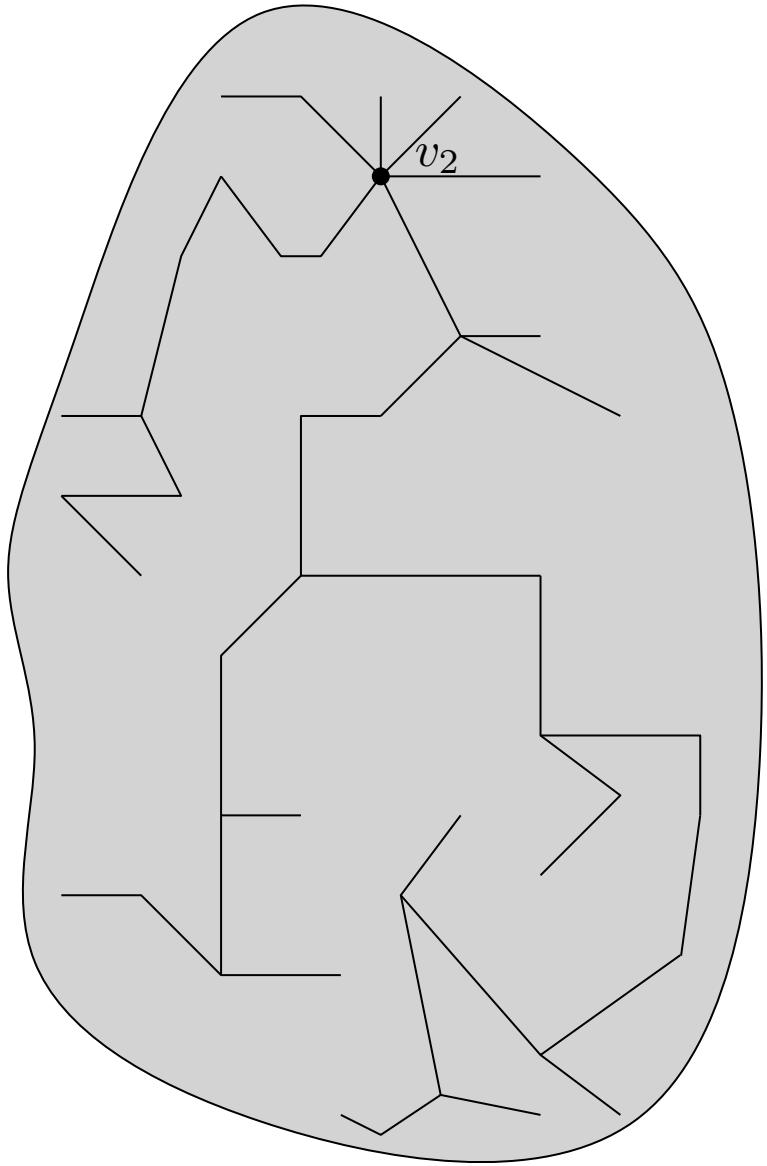
- for nodes u and v
- build shortest path trees from u and v
- compare them
- if they look similar, u and v are similar



Idea 1: Evaluating the View

Idea:

- for nodes u and v
- build shortest path trees from u and v
- compare them
- if they look similar, u and v are similar



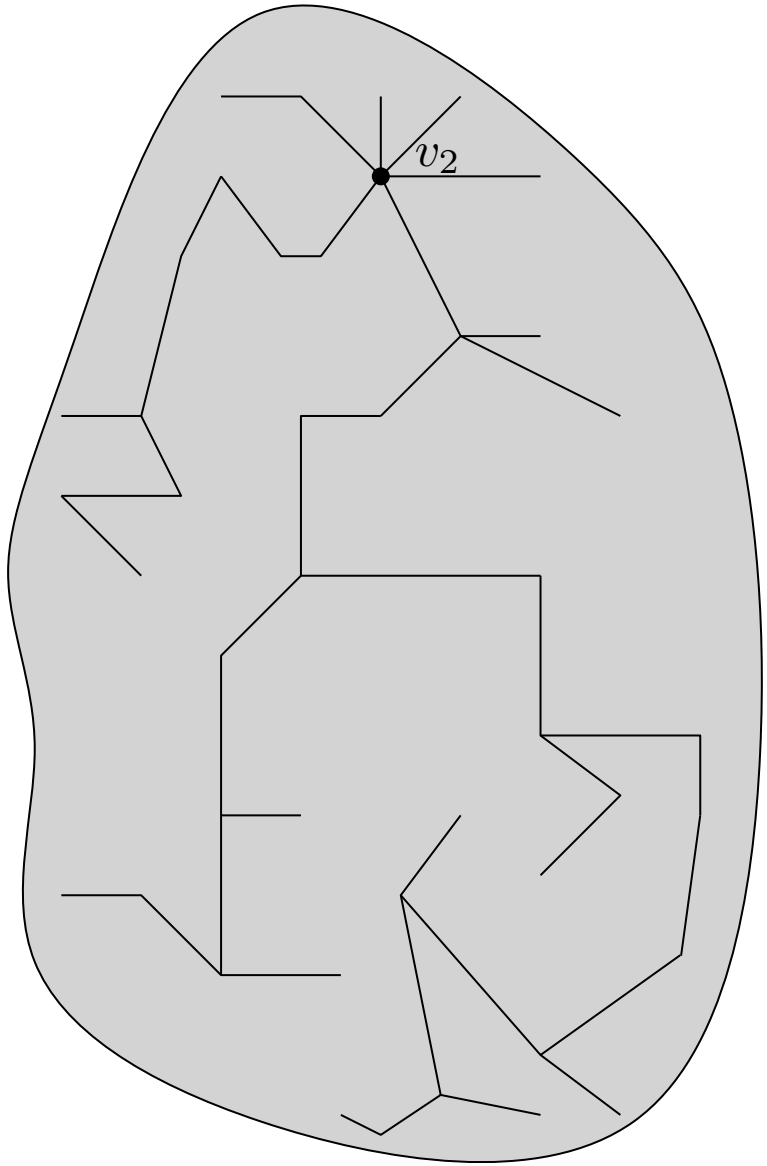
Idea 1: Evaluating the View

Idea:

- for nodes u and v
- build shortest path trees from u and v
- compare them
- if they look similar, u and v are similar

Problems:

- building two shortest path trees is time-consuming
- how exactly do we compare two trees?

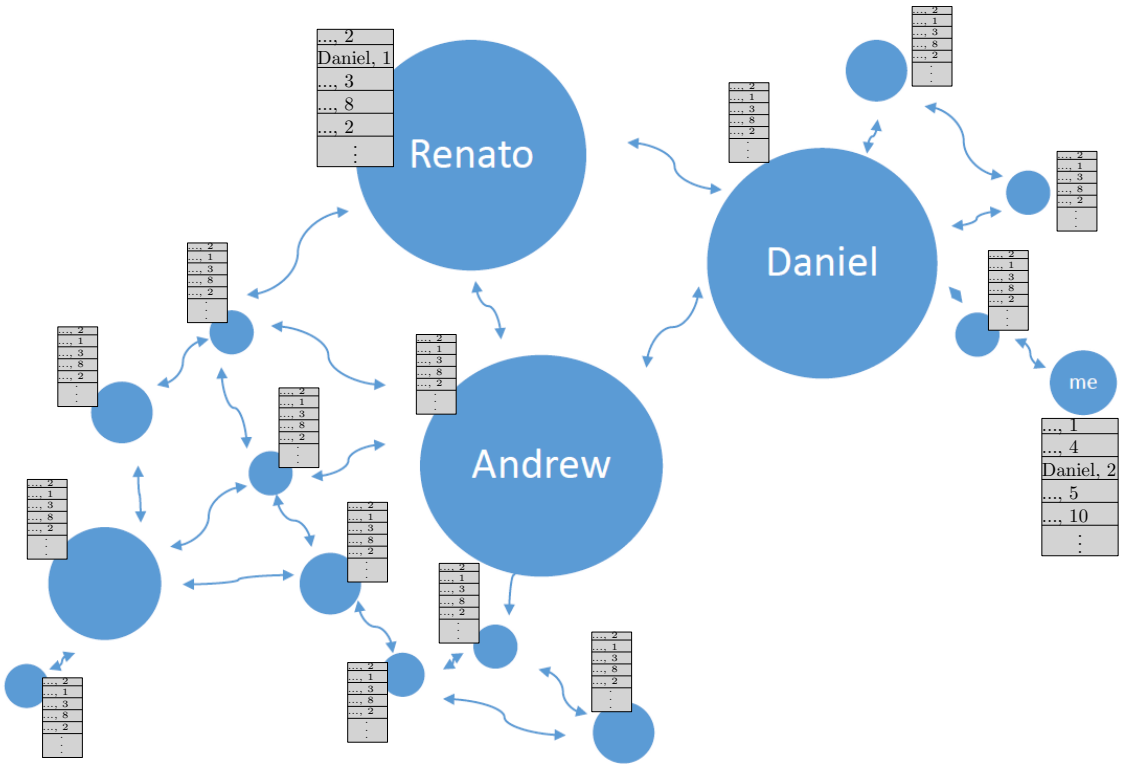


Idea 2: Efficient Computation

Main Idea:

- sample a subset of the nodes in the shortest-path trees using all-distances sketches: ADS(u) [Cohen97]
- each ADS(u) consists of a vector of (v, d_{vu}) pairs
- Expected ADS size is in $k \ln(n)$ per node

  n: number of nodes, k: constant (we use $k = 3$, resulting in label size ≈ 100)



Closeness similarity

- sufficient to compare ADS(u) and ADS(v)
  proof in the paper!
- the more they overlap,
  the more similar u and v are

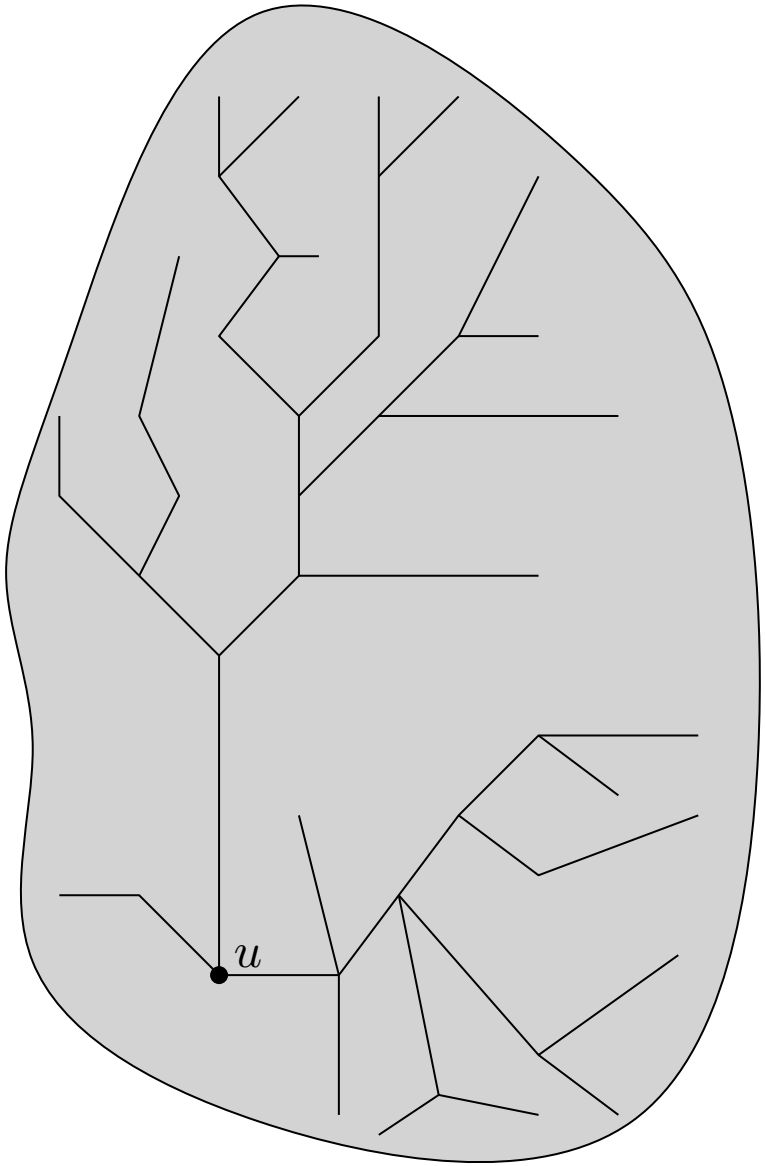
Definition

$$\text{Closeness}(u, v) = \frac{|\text{ADS}(u) \cap \text{ADS}(v)|}{|\text{ADS}(u) \cup \text{ADS}(v)|}$$

Remarks:

- easy to compute
- size of labels bounded

$\Rightarrow$ queries in microseconds



Closeness similarity

- sufficient to compare ADS(u) and ADS(v)
  proof in the paper!
- the more they overlap,
  the more similar u and v are

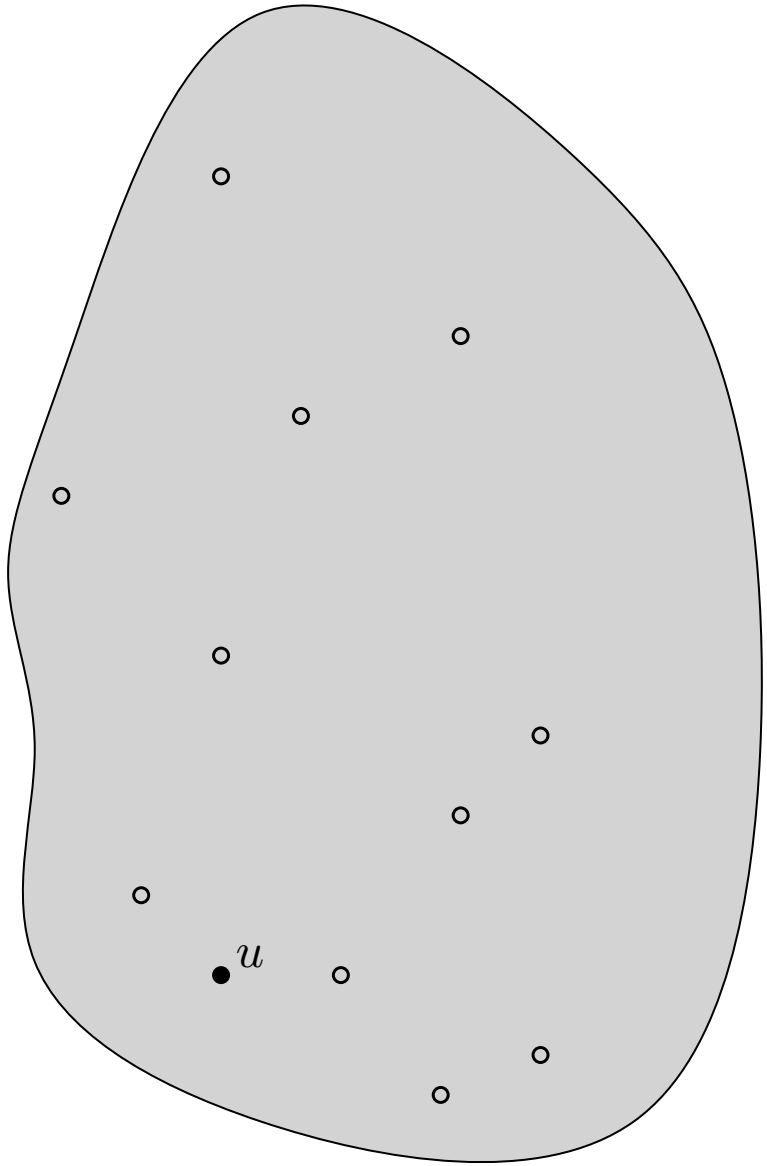
Definition

$$\text{Closeness}(u, v) = \frac{|\text{ADS}(u) \cap \text{ADS}(v)|}{|\text{ADS}(u) \cup \text{ADS}(v)|}$$

Remarks:

- easy to compute
- size of labels bounded

$\Rightarrow$ queries in microseconds



Closeness similarity

- sufficient to compare ADS(u) and ADS(v)
  proof in the paper!
- the more they overlap,
  the more similar u and v are

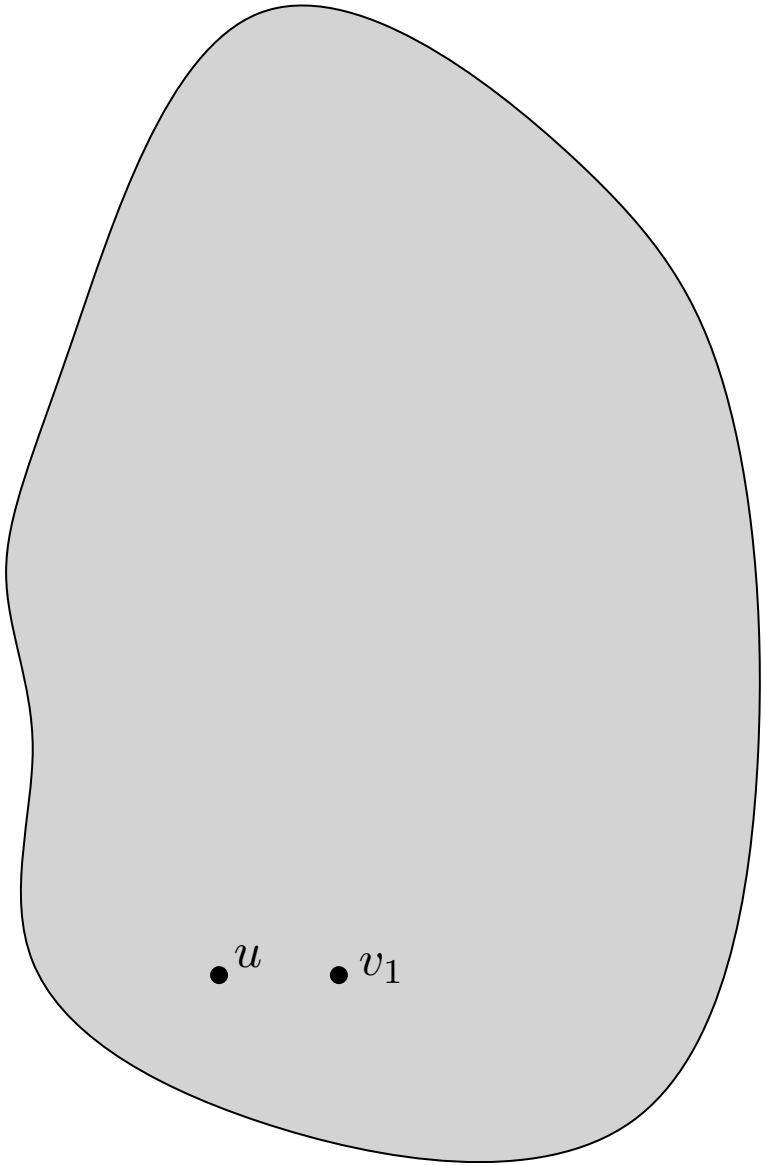
Definition

$$\text{Closeness}(u, v) = \frac{|\text{ADS}(u) \cap \text{ADS}(v)|}{|\text{ADS}(u) \cup \text{ADS}(v)|}$$

Remarks:

- easy to compute
- size of labels bounded

$\Rightarrow$ queries in microseconds



Closeness similarity

- sufficient to compare ADS(u) and ADS(v)
  proof in the paper!
- the more they overlap,
  the more similar u and v are

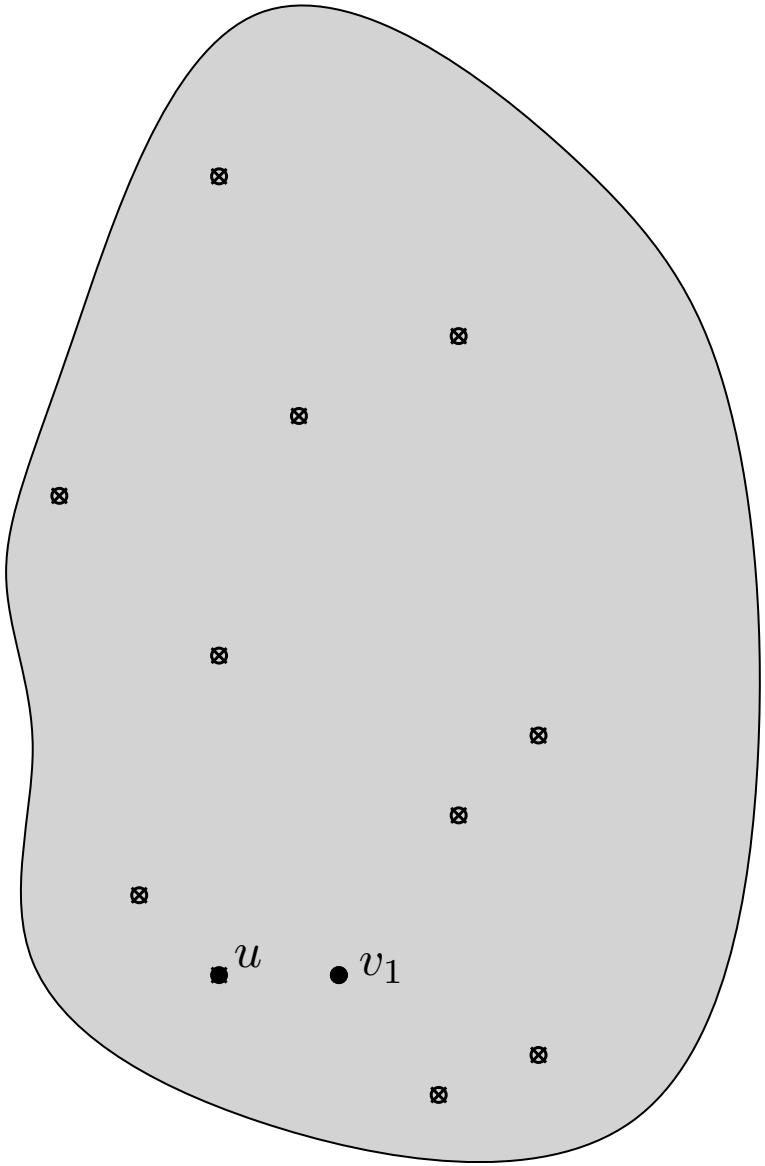
Definition

$$\text{Closeness}(u, v) = \frac{|\text{ADS}(u) \cap \text{ADS}(v)|}{|\text{ADS}(u) \cup \text{ADS}(v)|}$$

Remarks:

- easy to compute
- size of labels bounded

$\Rightarrow$ queries in microseconds



Closeness similarity

- sufficient to compare ADS(u) and ADS(v)
  proof in the paper!
- the more they overlap,
  the more similar u and v are

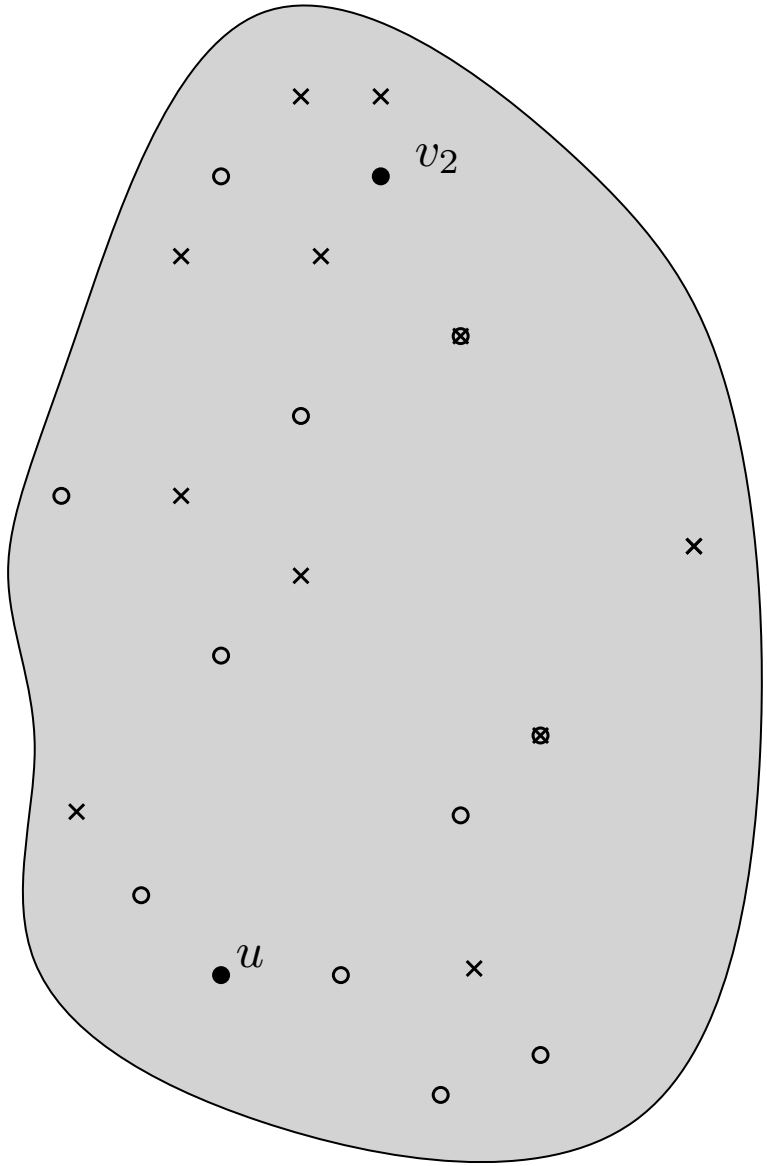
Definition

$$\text{Closeness}(u, v) = \frac{|\text{ADS}(u) \cap \text{ADS}(v)|}{|\text{ADS}(u) \cup \text{ADS}(v)|}$$

Remarks:

- easy to compute
- size of labels bounded

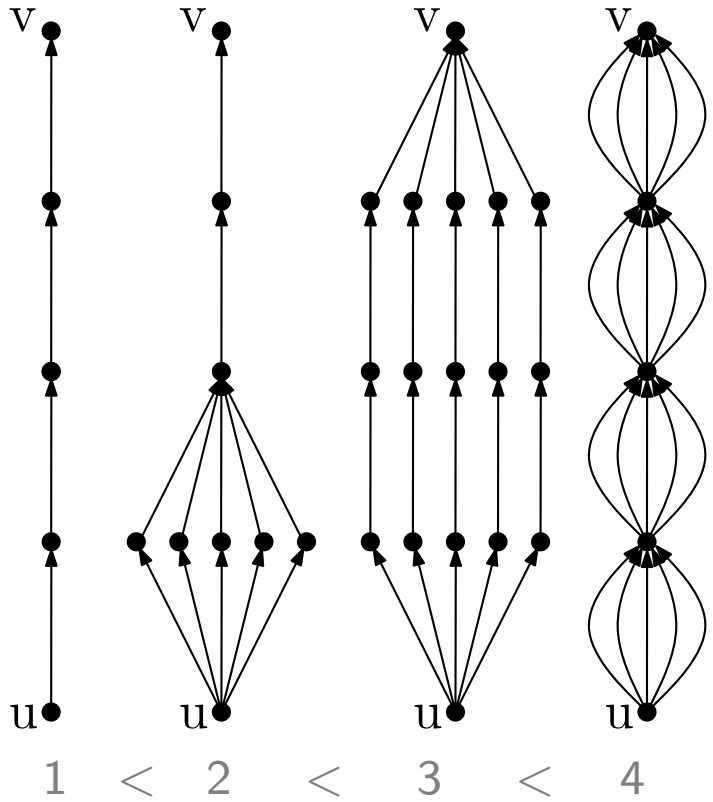
$\Rightarrow$ queries in microseconds



Idea 3: Dealing with Multiplicity

Problem:
distance does **not** account for multiple paths

Intuition:
similarity of a pair **should** increase with path multiplicity.



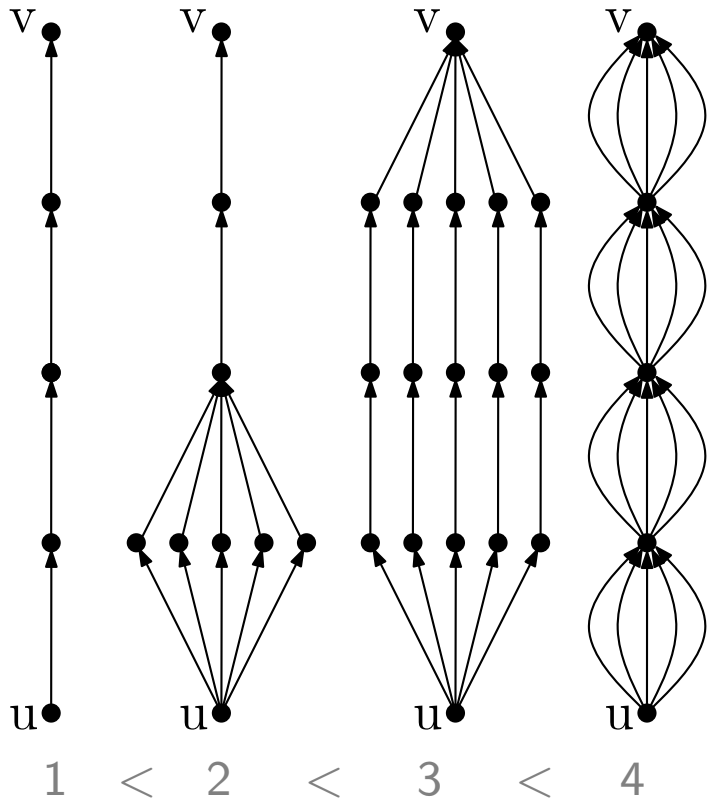
Idea 3: Dealing with Multiplicity

Problem:
distance does **not** account for multiple paths

Intuition:
similarity of a pair **should** increase with path multiplicity.

Idea:

- multiply each edge e by $-\ln(u_e)$
  u_e random number between 0 and 1
- repeat process multiple times
  we use 16
- highly connected pairs stay close with higher probability



Implementation: Compute multiple sketches, average over results.

Considered Networks

Networks:

- arXiv / DBLP co-authorship network
  - nodes: authors
  - undirected edges
  - edge weight: 1/(number of common papers)
  - ground truth: word similarity of title+abstracts or title+venue
- Twitter mention network, tweets from Dec. 2011
  - nodes: users
  - directed edges
  - edge weight: 1/(number of mentions)
  - ground truth: word similarity of tweets
- synthetic small world (SW) network [Kleinberg00]
  - toroidal grid with one additional edge per node
  - length of additional edges has length d with prob. 2^{-d} (in the grid)
  - undirected edges with unit weight
  - ground truth: L_1 distance

Key statistics and performance

Observations:

- we can handle large networks
- ADS computation times are reasonable
  not tuned at this point
- similarity can be evaluated within microseconds

graph	nodes [$\times 10^6$]	edges [$\times 10^6$]	prep.* [h:m]	label size	query* [μs]
arXiv	0.4	28.7	0:02	37.9	1.22
DBLP	1.1	9.2	0:02	39.1	1.64
twitter	29.6	603.9	8:05	101.6	3.51
smallworld	1.0	6.0	0:02	40.7	1.35

* single core of two 8-core Intel Xeon E-5-2690 2.90 GHz machine with 384 GiB of DDR3-1066 RAM.

Key statistics and performance

Observations:

- we can handle large networks
- ADS computation times are reasonable
  not tuned at this point
- similarity can be evaluated within microseconds

graph	nodes [$\times 10^6$]	edges [$\times 10^6$]	prep.* [h:m]	label size	query* [μs]
arXiv	0.4	28.7	0:02	37.9	1.22
DBLP	1.1	9.2	0:02	39.1	1.64
twitter	29.6	603.9	8:05	101.6	3.51
smallworld	1.0	6.0	0:02	40.7	1.35

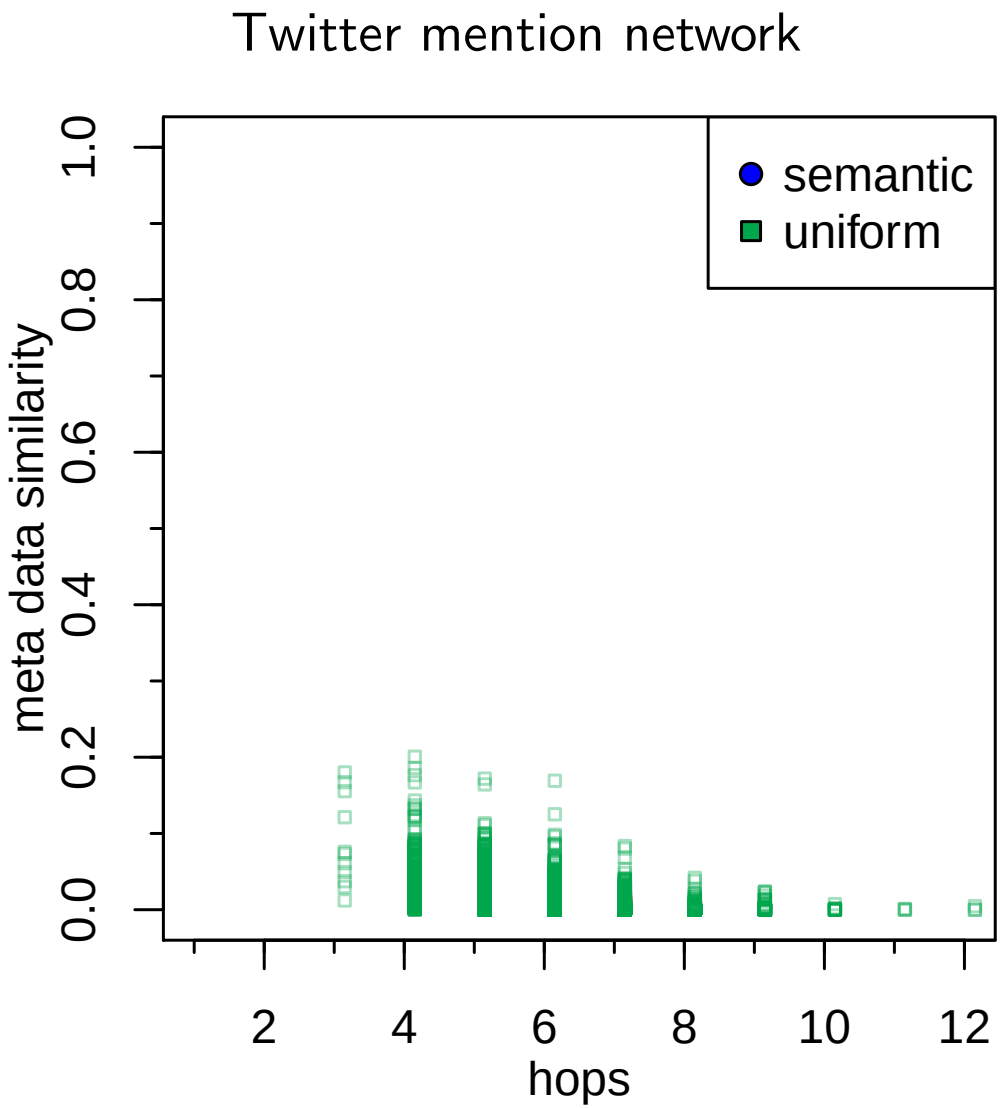
* single core of two 8-core Intel Xeon E-5-2690 2.90 GHz machine with 384 GiB of DDR3-1066 RAM.

Performance good, how about quality?

Which node pairs should we evaluate?

Uniform distribution:

- random picking of pairs
- often not very similar
- many far-away pairs



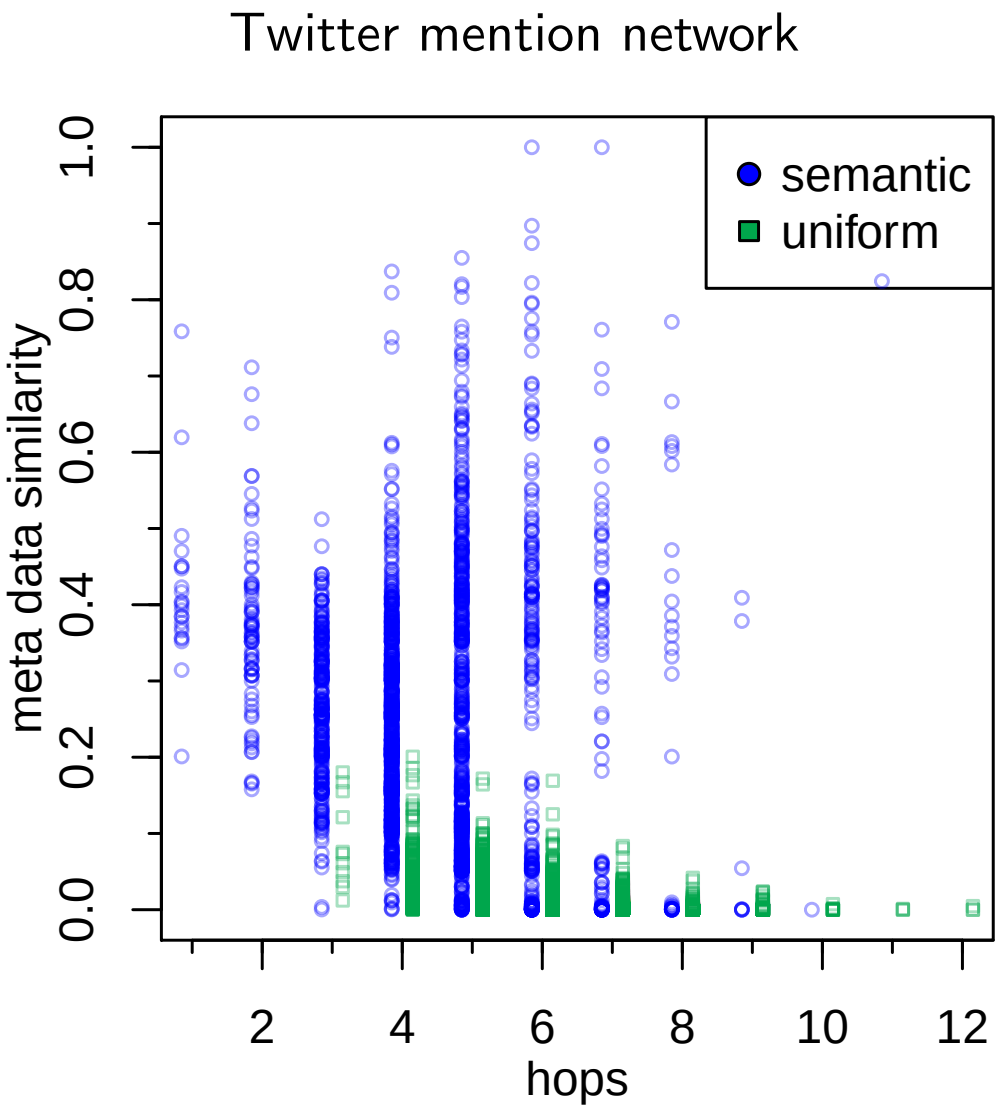
Which node pairs should we evaluate?

Uniform distribution:

- random picking of pairs
- often not very similar
- many far-away pairs

Semantic distribution:

- uniform over ground truth (as much as possible)
- many similar pairs
- often not very far away



How to evaluate?

Similarity evaluation

- evaluate similarity measure for each pair
- rank the pairs according to the similarity measure and metadata similarity
- compare rankings using Spearman's rank correlation
- correlation values of 1 are perfect, 0 is random.

Results

measure	arXiv	DBLP	Twitter	SW
Adamic-Adar				
hops				
distance				
RWR-0.75				
RWR-0.50				
RWR-0.25				
RWR-0.00				
Closeness				
Closeness REL				

Results

measure	arXiv	DBLP	Twitter	SW
Adamic-Adar	0.626	0.746	0.548	0.000
hops	**0.752**	0.748	0.169	0.767
distance	0.590	0.634	−0.140	0.767
RWR-0.75	—	0.734	—	0.286
RWR-0.50	—	0.737	—	0.617
RWR-0.25	—	0.740	—	0.791
RWR-0.00	—	0.500	—	**0.915**
Closeness	0.641	0.742	0.613	0.609
Closeness REL	0.634	**0.752**	**0.649**	0.808

Results

Observations:

- global approaches perform better
- Closeness REL
  - superior in twitter and DBLP
  - competitive in other networks
  - much faster queries after preprocessing

measure	arXiv	DBLP	Twitter	SW
Adamic-Adar	0.626	0.746	0.548	0.000
hops	**0.752**	0.748	0.169	0.767
distance	0.590	0.634	−0.140	0.767
RWR-0.75	—	0.734	—	0.286
RWR-0.50	—	0.737	—	0.617
RWR-0.25	—	0.740	—	0.791
RWR-0.00	—	0.500	—	**0.915**
Closeness	0.641	0.742	0.613	0.609
Closeness REL	0.634	**0.752**	**0.649**	0.808

Conclusion

Results

- **Global similarity computation at scale**
  - Closeness similarity
    in the paper: a lot more general than in this talk
  - ADS labels
  - REL

Conclusion

Results

- **Global similarity computation at scale**
  - Closeness similarity
    in the paper: a lot more general than in this talk
  - ADS labels
  - REL

Future work

- more experiments with different distance decay and node weight functions
- compute ADS online-dynamically
- apply these techniques to community detection

Conclusion

Results

- **Global similarity computation at scale**
  - Closeness similarity
    in the paper: a lot more general than in this talk
  - ADS labels
  - REL

Future work

- more experiments with different distance decay and node weight functions
- compute ADS online-dynamically
- apply these techniques to community detection

Thank you!

Appendix

Closeness similarity

Definition of Closeness similarity

$$J_{\alpha,\beta}(v,u) = \frac{\sum_w \alpha(\max\{\delta_{vw}, \delta_{uw}\})\beta(w)}{\sum_w \alpha(\min\{\delta_{vw}, \delta_{uw}\})\beta(w)}.$$

Closeness similarity

Definition of Closeness similarity

$$J_{\alpha,\beta}(v,u) = \frac{\sum_w \alpha(\max\{\delta_{vw},\delta_{uw}\})\beta(w)}{\sum_w \alpha(\min\{\delta_{vw},\delta_{uw}\})\beta(w)}.$$

$\alpha(x) \geq 0$, monotone non-increasing

$\beta(w) \geq 0$

distance

Closeness similarity

Definition of Closeness similarity

$$J_{\alpha,\beta}(v,u) = \frac{\sum_w \alpha(\max\{\delta_{vw}, \delta_{uw}\})\beta(w)}{\sum_w \alpha(\min\{\delta_{vw}, \delta_{uw}\})\beta(w)}.$$

$\alpha(x) \geq 0$, monotone non-increasing

$\beta(w) \geq 0$

distance

- very general definition
  - arbitrary distance decay function $\alpha(\cdot)$ and node weight function $\beta(\cdot)$
  - generalizes and extends many local measures
  - extends Adamic-Adar for $\alpha(x) = 1/(1+x)$ and $\beta(w) = 1/\log|\Gamma(w)|$, where $|\Gamma(w)|$ is the degree of w.
  - considers the whole graph, hence computationally expensive

Closeness similarity

Definition of Closeness similarity

$$J_{\alpha,\beta}(v,u) = \frac{\sum_w \alpha(\max\{\delta_{vw},\delta_{uw}\})\beta(w)}{\sum_w \alpha(\min\{\delta_{vw},\delta_{uw}\})\beta(w)}.$$

$\alpha(x) \geq 0$, monotone non-increasing

$\beta(w) \geq 0$

distance

- very general definition
  - arbitrary distance decay function $\alpha(\cdot)$ and node weight function $\beta(\cdot)$
  - generalizes and extends many local measures
  - extends Adamic-Adar for $\alpha(x) = 1/(1+x)$ and $\beta(w) = 1/\log|\Gamma(w)|$, where $|\Gamma(w)|$ is the degree of w.
  - considers the whole graph, hence computationally expensive

- we can estimate Closeness similarity efficiently using ADS labels
  - considers only nodes in the ADS labels
  - the root of the expected square error is in $O(1/\sqrt{k})$
  - distance decay function $\alpha(\cdot)$ and node weight function $\beta(\cdot)$ can be chosen to fit the application
  - more results in the paper, including the full estimator, bounds on estimation and computation, how to improve bounds, etc ...

All-distance sketch (ADS) construction

1. **Assign a rank** $r(u) \in [0, 1]$ to each node u in the network
2. **Build ADS label**
   - Definition:
     $\text{ADS}(v) := \{(u, d_{vu}) | r(u)$ is one of the k smallest $r(\cdot)$ within d_{vu} of $v\}$
   - Naïve approach: Compute each label separately
   - More efficient: Add each node to respective labels, start with low $r(\cdot)$ nodes
     - still n Dijkstra executions, however:
     - prune Dijkstra whenever a node is not added
     - very few nodes per Dijkstra considered for nodes with high $r(\cdot)$ values
3. **Sort** each label by node id

Overall complexity: $\mathcal{O}(km \ln^2(n))$

ADS distance as similarity:

$$\text{ADS distance}(u, v) = \min_{\substack{(w, d_{\cdot w}) \in \\ \text{ADS}(u) \cap \text{ADS}(v)}} d_{uw} + d_{vw}$$

ADS distance is an upper bound on the actual distance.

Semantic distribution

measure	arXiv	DBLP	Twitter	SW
Adamic-Adar	0.626	0.746	0.548	0.000
hops	**0.752**	0.748	0.169	0.767
RWR-0.75	—	0.734	—	0.286
RWR-0.50	—	0.737	—	0.617
RWR-0.25	—	0.740	—	0.791
RWR-0.00	—	0.500	—	**0.915**
distance	0.590	0.634	−0.140	0.767
ADS dist	0.566	0.637	−0.127	0.671
ADS dist REL	0.614	0.584	−0.155	0.865
Closeness	0.641	0.742	0.613	0.609
Closeness REL	0.634	**0.752**	**0.649**	0.808

Uniform distribution

measure	arXiv	DBLP	Twitter	SW
Adamic-Adar	0.097	0.034	0.107	0.000
hops	0.350	0.221	0.536	0.623
distance	**0.470**	**0.319**	0.191	0.623
ADS dist	0.462	0.318	0.196	0.519
ADS dist REL	0.419	0.314	0.242	**0.769**
Closeness	0.039	0.015	0.461	0.413
Closeness REL	0.063	0.034	**0.612**	0.666

Hop-based distribution

measure	arXiv	DBLP	Twitter	SW
Adamic-Adar	0.570	0.457	0.420	0.486
hops	0.645	0.468	**0.678**	0.831
distance	**0.648**	**0.507**	0.447	0.831
ADS dist	0.617	0.497	0.448	0.839
ADS dist REL	0.512	0.454	0.471	0.947
Closeness	0.379	0.249	0.518	0.877
Closeness REL	0.404	0.320	0.637	**0.949**