Social Influence, Community Kernels, and Structural Holes

Jie Tang
Tsinghua University, China

Collaborate with
Jimeng Sun (*IBM TJ Watson*)
Jiawei Han and Chi Wang (*UIUC*)
Liaoruo Wang and John Hopcroft (*Cornell*)
Zi Yang (*CMU*), Lu Liu and Tiancheng Lou (*THU*)

Social Networks

- >**900 million** users
- the **3rd** largest "Country" in the world
- More visitors than Google

- >**500 million** users

- More than **5 billion** images

- 2009, **2 billion** tweets per quarter
- 2010, **4 billion** tweets per quarter
- 2011, **25 billion** tweets per quarter

- 2012, **300 million** users, 300% yearly increase

- Pinterest, with a traffic higher than Twitter and Google

Social Networks

- >**900 million** users
- the **3rd** largest "Country" in the world
- More visitors than Google

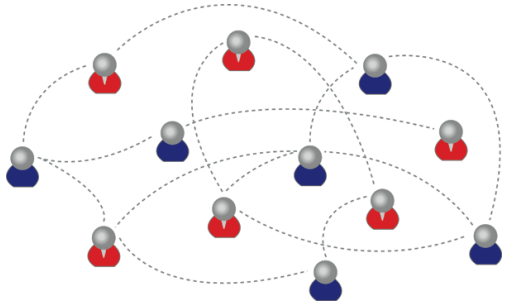
Social networks already become a bridge to connect our daily **physical** life and the **virtual** web space

O2O!

- 2012, **300 million** users, 300% yearly increase

- Pinterest, with a traffic higher than Twitter and Google

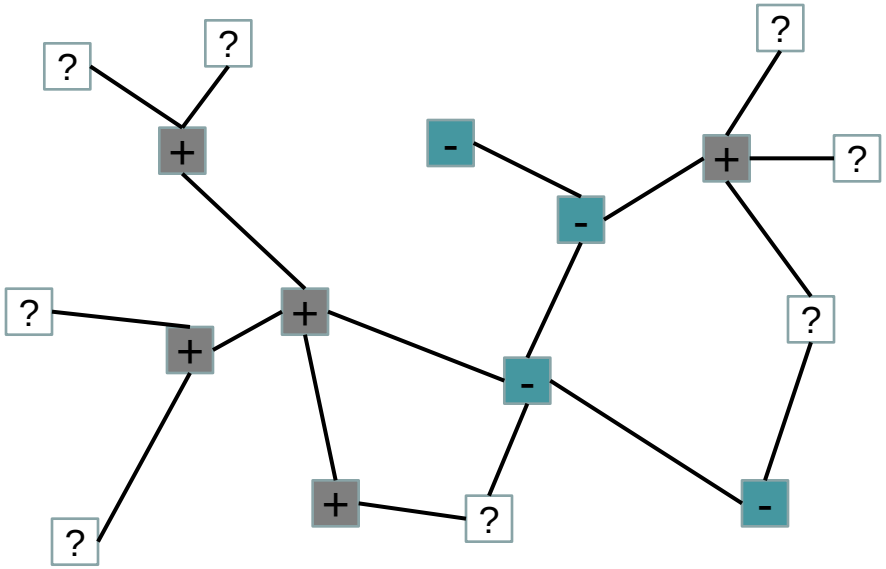
What is a social network?



- A social network is:
  - a **graph** made up of :
  - a set of **individuals**, called “nodes”, and
  - tied by one or more **interdependency**, such as friendship, called “edges”.

What are the fundamentally new things in social networks?

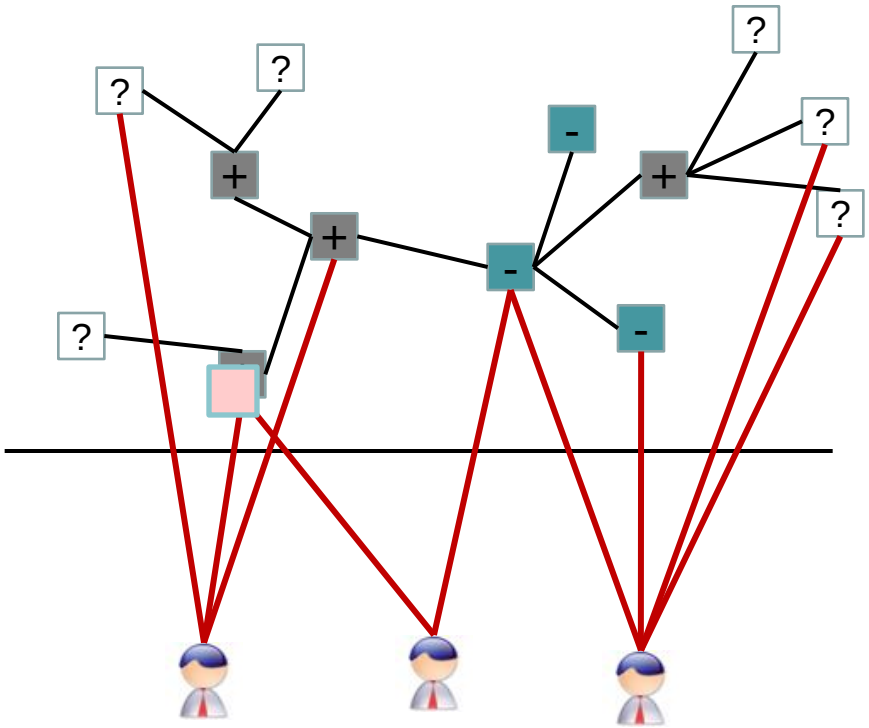
Web 1.0



(1) links between data points
(2) the network can be heterogeneous

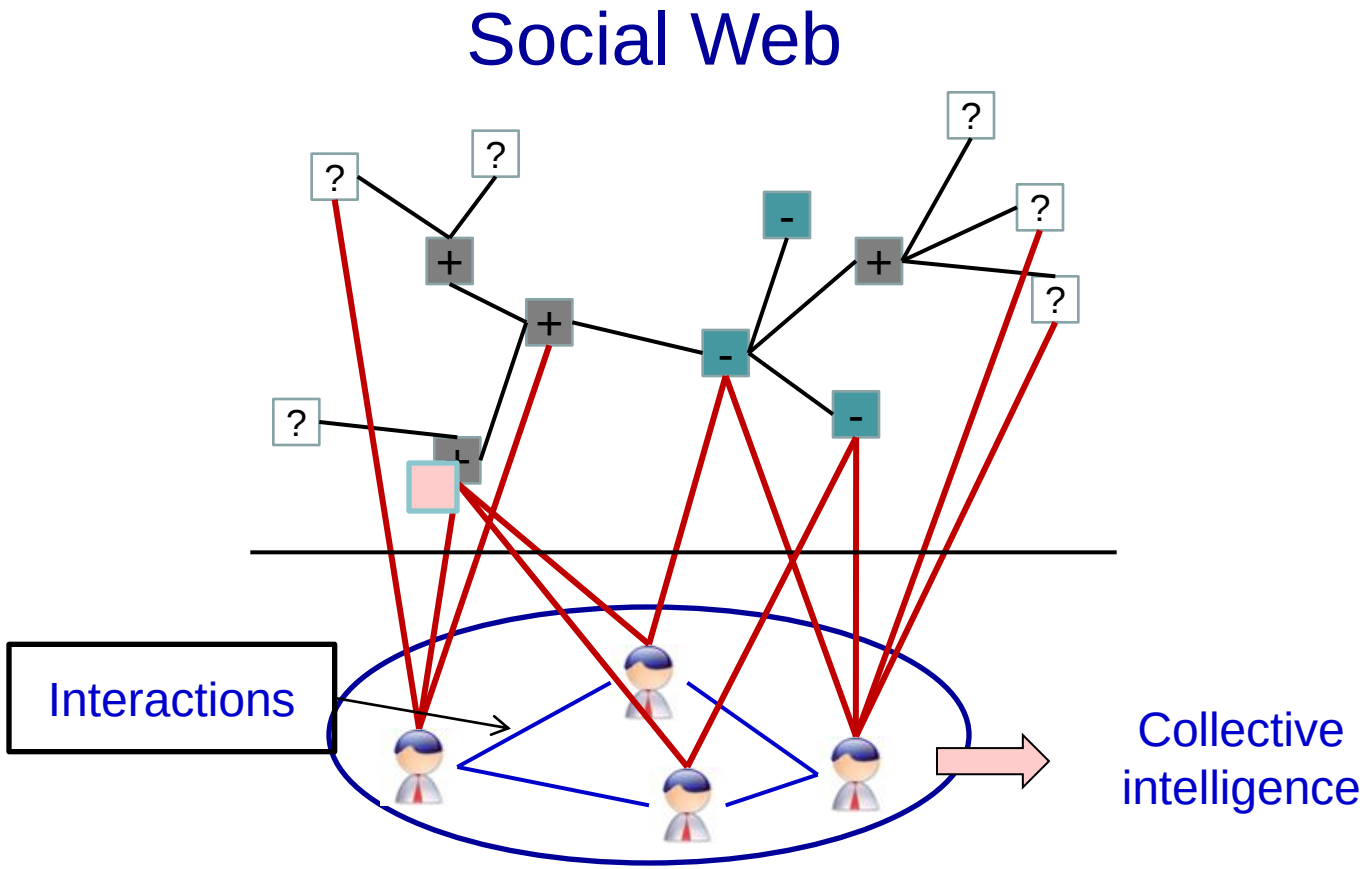
What are the fundamentally new things in social networks?

Collaborative Web



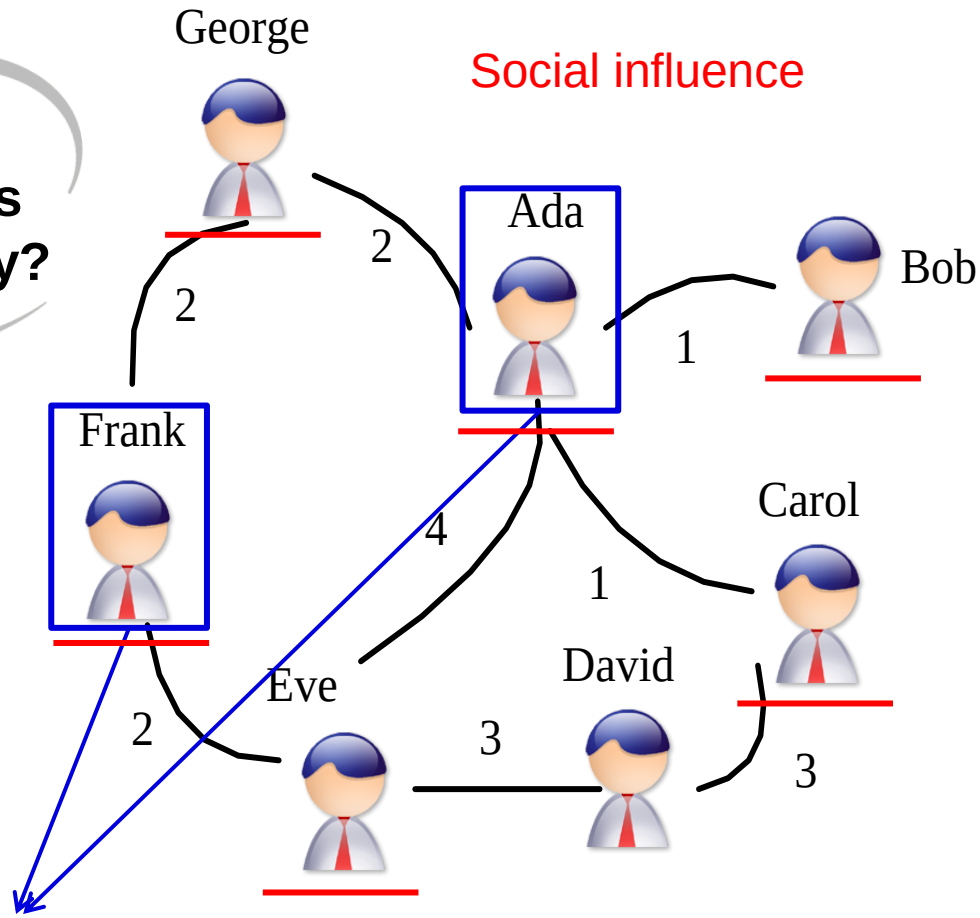
(1) personalized learning
(2) collaborative filtering

What are the fundamentally new things in social networks?



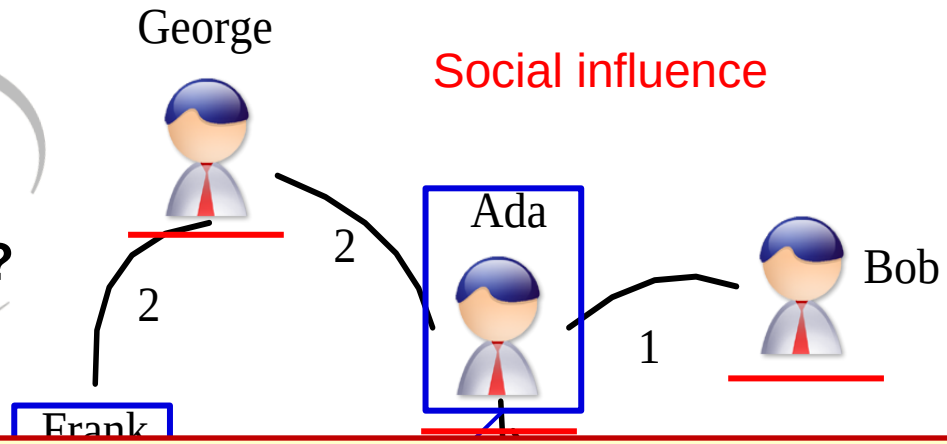
(1) interactions
(2) information diffusion

Example—opinion leaders



Find *K* nodes (users) in a social network that could maximize the spread of influence (Domingos, 01; Richardson, 02; Kempe, 03)

Example—opinion leaders

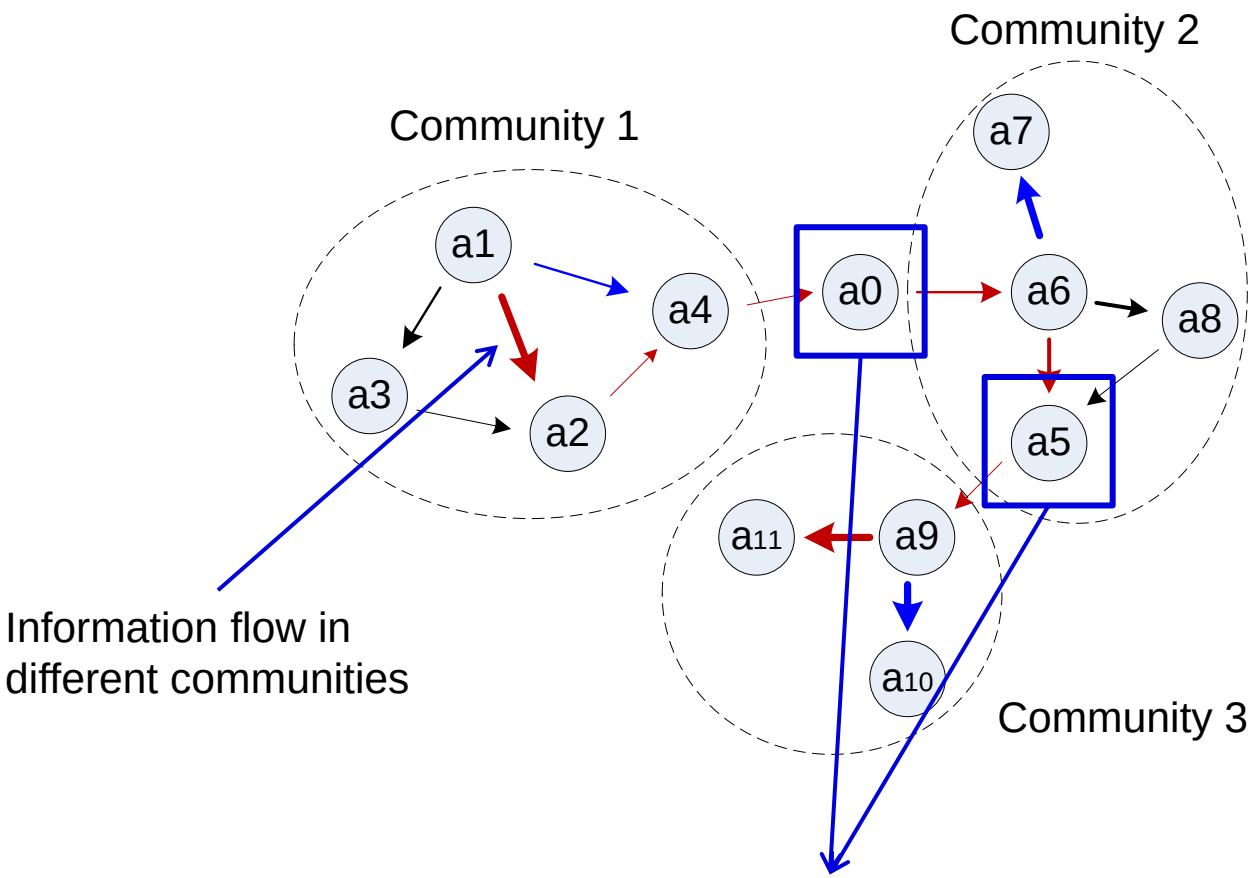


Questions:

- How to quantify the strength of social influence between users?
- How to predict users' behaviors over time?

spread of influence (Domingos, 01; Richardson, 02; Kempe, 03)

Example—structural holes



Structural hole users control the information flow between different communities (Burt, 92; Podolny, 97; Ahuja, 00; Kleinberg, 08)

Motivation

- Modeling the influence between users and the network structural formation becomes a very important issue and can benefit many real applications
  - Advertising
  - Social recommendation
  - Marketing
  - Social security
  - …

Social Influence Analysis

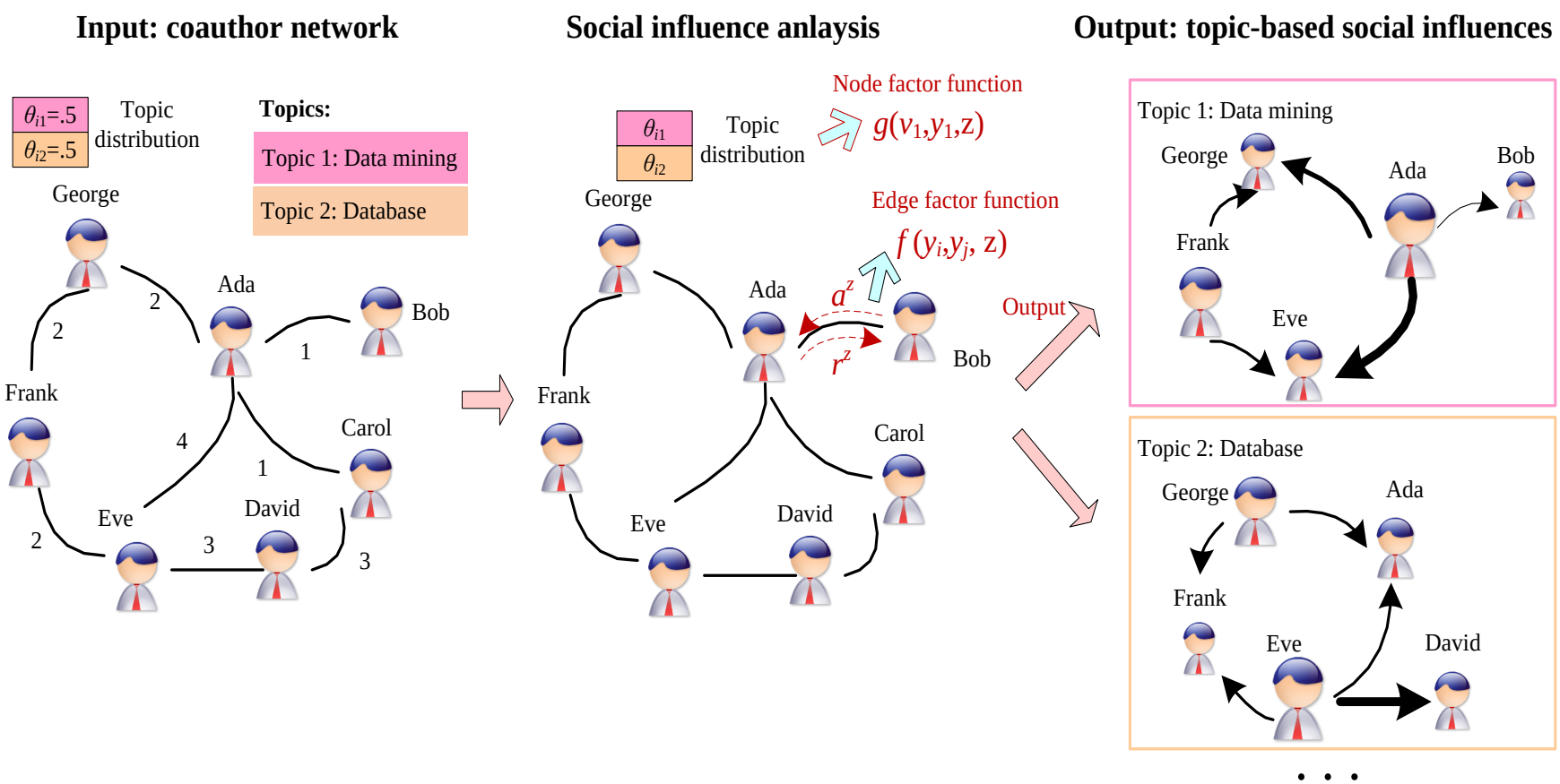
- Jie Tang, Jimeng Sun, Chi Wang, and Zi Yang. Social Influence Analysis in Large-scale Networks. **SIGKDD 2009**. pp. 807-816.

- Lu Liu, Jie Tang, Jiawei Han, and Shiqiang Yang. Learning Influence from Heterogeneous Social Networks. **Data Mining and Knowledge Discovery**. (to appear)

- Lu Liu, Jie Tang, Jiawei Han, Meng Jiang, and Shiqiang Yang. Mining Topic-Level Influence in Heterogeneous Networks. **CIKM 2010**. pp. 199-208.

Learning topic-based influence between users

- Social network -> Topical influence network



How two persons Influence each other?

How a person influence a social community?

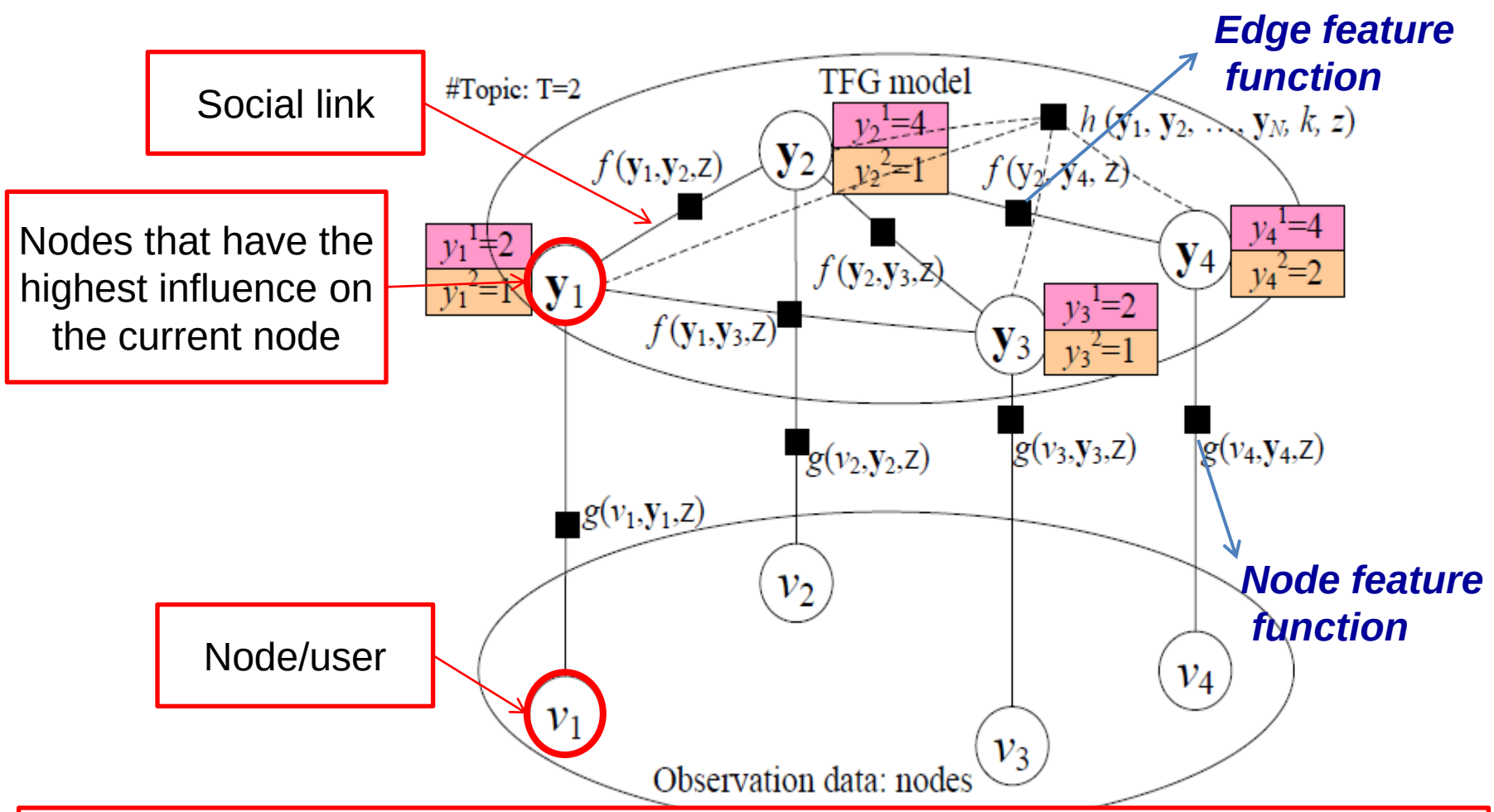
Several key challenges:

- How to differentiate the social influences from different angles (topics)?
- How to incorporate different information (e.g., topic distribution and network structure) into a unified model?
- How to estimate the model on real-large networks?

Our Solution: Topical Affinity Propagation

- Topical Affinity Propagation
  - Topical Factor Graph model
  - Efficient learning algorithm
  - Distributed implementation

Topical Factor Graph (TFG) Model



The problem is cast as identifying which node has the **highest probability** to **influence** another node on a **specific topic** along with the edge.

Topical Factor Graph (TFG)

Objective function:

$$P(\mathbf{v}, \mathbf{Y}) = \frac{1}{Z} \prod_{k=1}^{N} \prod_{z=1}^{T} \boxed{h(\mathbf{y}_1, \ldots, \mathbf{y}_N, k, z)}$$

$$\prod_{i=1}^{N} \prod_{z=1}^{T} \boxed{g(v_i, \mathbf{y}_i, z)} \prod_{e_{kl} \in E} \prod_{z=1}^{T} \boxed{f(\mathbf{y}_k, \mathbf{y}_l, z)}$$

1. How to define?

2. How to optimize?

- The learning task is to find a configuration for all $\{y_i\}$ to maximize the joint probability.

How to define (topical) feature functions?

– Node feature function

similarity

$$g(v_i, \mathbf{y}_i, z) = \begin{cases} \dfrac{\boxed{w^z_{iy^z_i}}}{\sum_{j \in NB(i)} (w^z_{ij} + w^z_{ji})} & y^z_i \neq i \\ \dfrac{\sum_{j \in NB(i)} w^z_{ji}}{\sum_{j \in NB(i)} (w^z_{ij} + w^z_{ji})} & y^z_i = i \end{cases}$$

– Edge feature function

$$f(y_i, y_j) = \begin{cases} w[v_i \sim v_j] & y_i = y_j \\ 1 - w[v_i \sim v_j] & y_i \neq y_j \end{cases}$$

or simply binary

– Global feature function

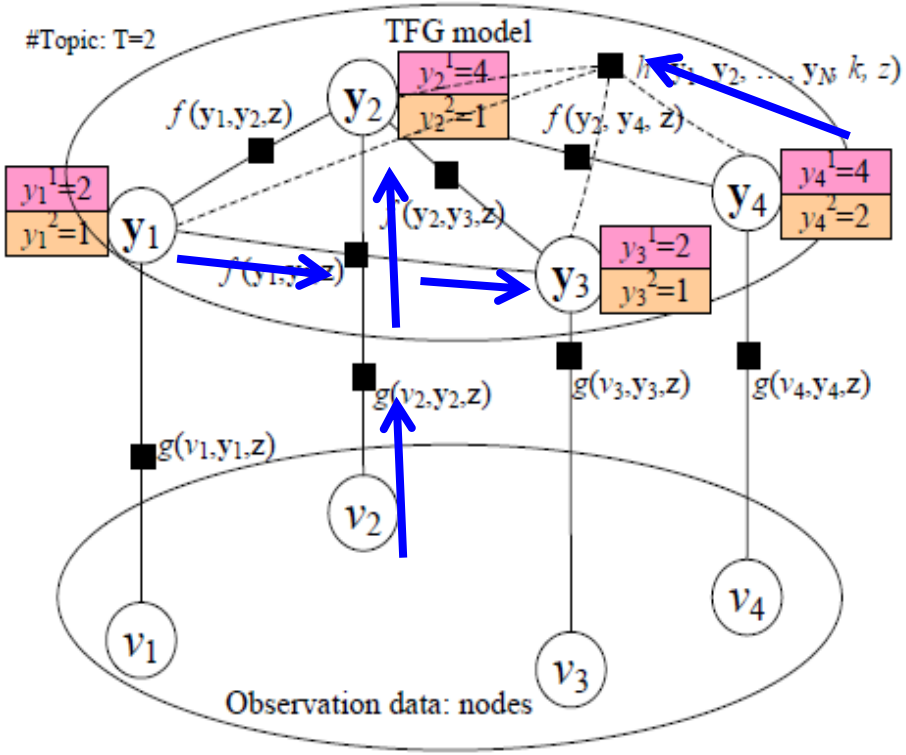
$$h(\mathbf{y}_1, \ldots, \mathbf{y}_N, k, z) = \begin{cases} 0 & \text{if } y^z_k = k \text{ and } y^z_i \neq k \text{ for all } i \neq k \\ 1 & \text{otherwise.} \end{cases}$$

Model Learning Algorithm

Sum-product:

$$m_{y_i \to f_{ij}(y_i)} = \prod_{f' \sim y_i \setminus f_{ij}} m_{f' \to y_i(y_i)}$$

$$m_{f_{ij} \to y_i(y_i)} = \sum_{\sim\{y_i\}} \left(\prod_{y' \sim f \setminus y_i} f(y_i, y') m_{y' \to f_{ij}(y')} \right)$$



- Low efficiency!
- Not easy for distributed learning!

New TAP Learning Algorithm

1. Introduce two new variables *r* and *a, to* replace the original message *m*.

2. Design new update rules:

$$m_{ij} \rightarrow r_{ij}^z = b_{ij}^z - \max_{k \in NB(j)} \{b_{ik}^z + a_{ik}^z\}$$

$$m_{ij} \rightarrow a_{jj}^z = \max_{k \in NB(j)} \min\{r_{kj}^z, 0\}$$

$$a_{ij}^z = \min(\max\{r_{jj}^z, 0\}, -\min\{r_{jj}^z, 0\}$$

$$- \max_{k \in NB(j) \setminus \{i\}} \min\{r_{kj}^z, 0\}), i \in NB(j)$$

The TAP Learning Algorithm

Input: $G = (V, E)$ and topic distributions $\{\theta_v\}_{v \in V}$
Output: topic-level social influence graphs $\{G_z = (V_z, E_z)\}^T$

Calculate the node feature function $g(v_i, \mathbf{y}_i, z)$;
Calculate b_{ij}^z according to Eq. 8;
Initialize all $\{r_{ij}^z\} \leftarrow 0$;
repeat
  foreach *edge-topic pair* (e_{ij}, z) **do**
    Update r_{ij}^z according to Eq. 5;
  end
  foreach *node-topic pair* (v_j, z) **do**
    Update a_{jj}^z according to Eq. 6;
  end
  foreach *edge-topic pair* (e_{ij}, z) **do**
    Update a_{ij}^z according to Eq. 7;
  end
until *convergence*;
foreach *node* v_t **do**
  foreach *neighboring node* $s \in NB(t) \cup \{t\}$ **do**
    Compute μ_{st}^z according to Eq. 9;
  end
end
Generate $G_z = (V_z, E_z)$ for every topic z according to $\{\mu_{st}^z\}$;

$$b_{ij}^z = \log \frac{g(v_i, \mathbf{y}_i, z)|_{y_i^z = j}}{\sum_{k \in NB(i) \cup \{i\}} g(v_i, \mathbf{y}_i, z)|_{y_i^z = k}}$$

$$r_{ij}^z = b_{ij}^z - \max_{k \in NB(j)} \{b_{ik}^z + a_{ik}^z\}$$

$$a_{jj}^z = \max_{k \in NB(j)} \min\{r_{kj}^z, 0\}$$

$$a_{ij}^z = \min(\max\{r_{jj}^z, 0\}, -\min\{r_{jj}^z, 0\} - \max_{k \in NB(j) \setminus \{i\}} \min\{r_{kj}^z, 0\}), i \in NB(j)$$

$$\mu_{st}^z = \frac{1}{1 + e^{-(r_{ts}^z + a_{ts}^z)}}$$

Distributed TAP Learning

- Map-Reduce
  - Map: (key, value) pairs
    - $e_{ij}/a_{ij} \rightarrow e_{i*}/a_{ij}$; $e_{ij}/b_{ij} \rightarrow e_{i*}/b_{ij}$; $e_{ij}/r_{ij} \rightarrow e_{*j}/r_{ij}$.
  - Reduce: (key, value) pairs
    - e_{ij} / * → new r_{ij}; e_{ij}/* → new a_{ij}
- For the global feature function

THEOREM 1. *If the global feature function h can be factorized into $h = \prod_{k=1}^{N} h_k$, for every $i \in \{1, \ldots, N\}, y_i \neq k, y_i' \neq k, h_k(y_1, \ldots, y_i, \ldots, y_N) = h_k(y_1, \ldots, y_i', \ldots, y_N)$, then the message passing update rules can be simplified to influence update rules.* ■

Experiment

- Data set: (http://arnetminer.org/lab-datasets/soinf/)

Data set	#Nodes	#Edges
Coauthor	640,134	1,554,643
Citation	2,329,760	12,710,347
Film	18,518 films 7,211 directors 10,128 actors 9,784 writers	142,426

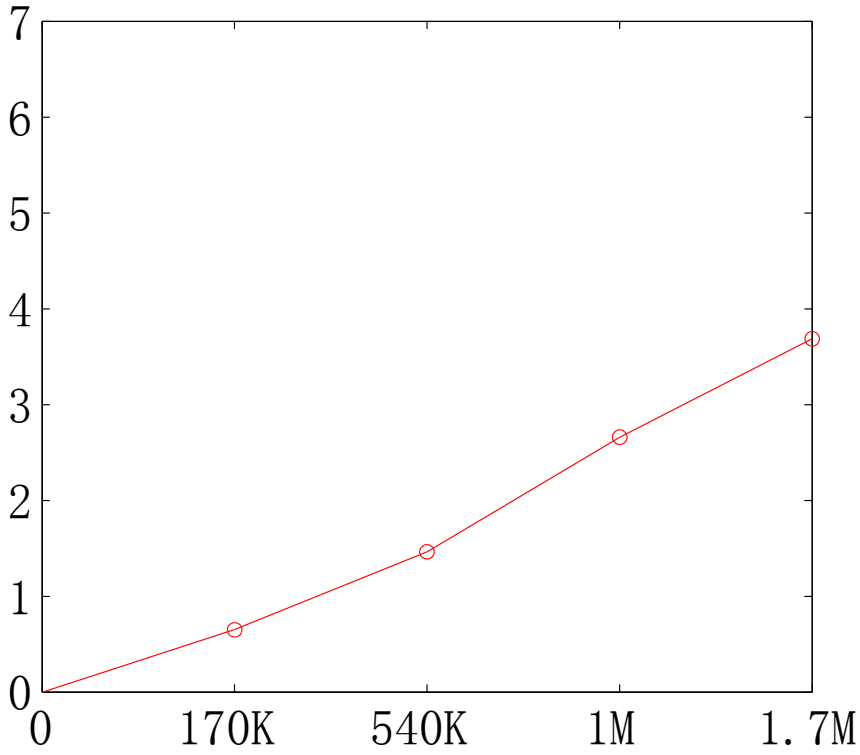
- Evaluation measures
  - CPU time
  - Case study
  - Application

Scalability Performance

Table 2: Scalability performance of different methods on real data sets. >10hr means that the algorithm did not terminate when the algorithm runs more than 10 hours.

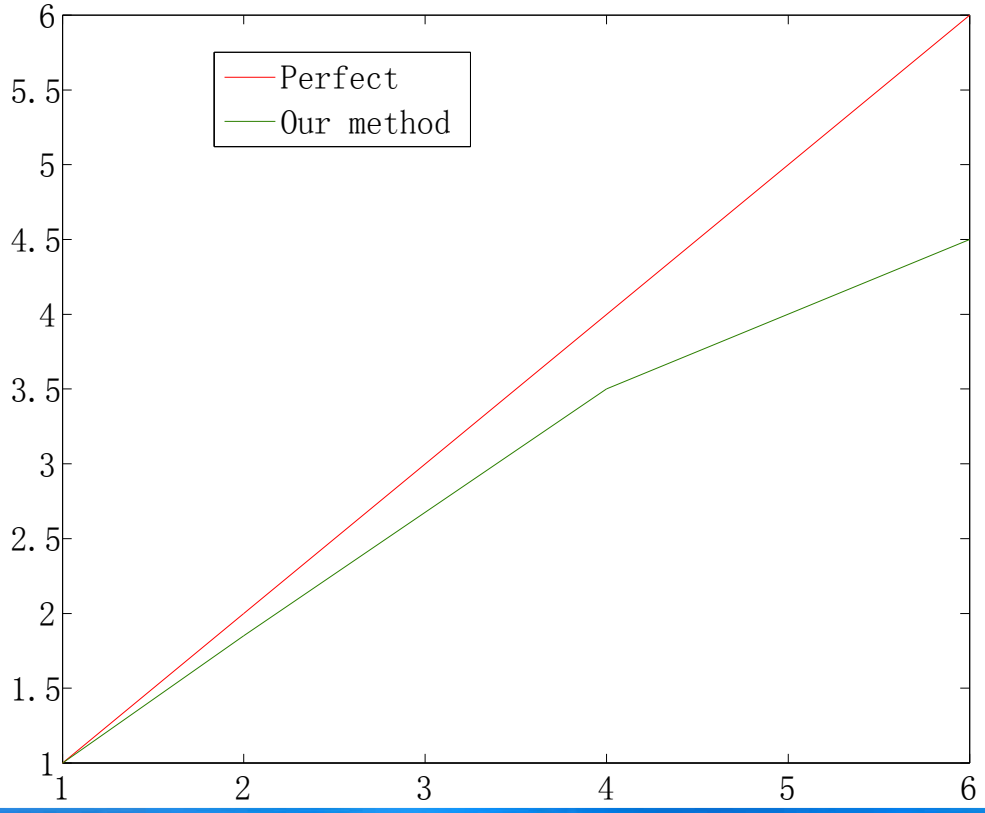
Methods	Citation	Coauthor	Film
Sum-Product	N/A	>10hr	1.8 hr
Basic TAP Learning	>10hr	369s	**57s**
Distributed TAP Learning	**39.33m**	**104s**	148s

Speedup results



Speedup vs. Dataset size

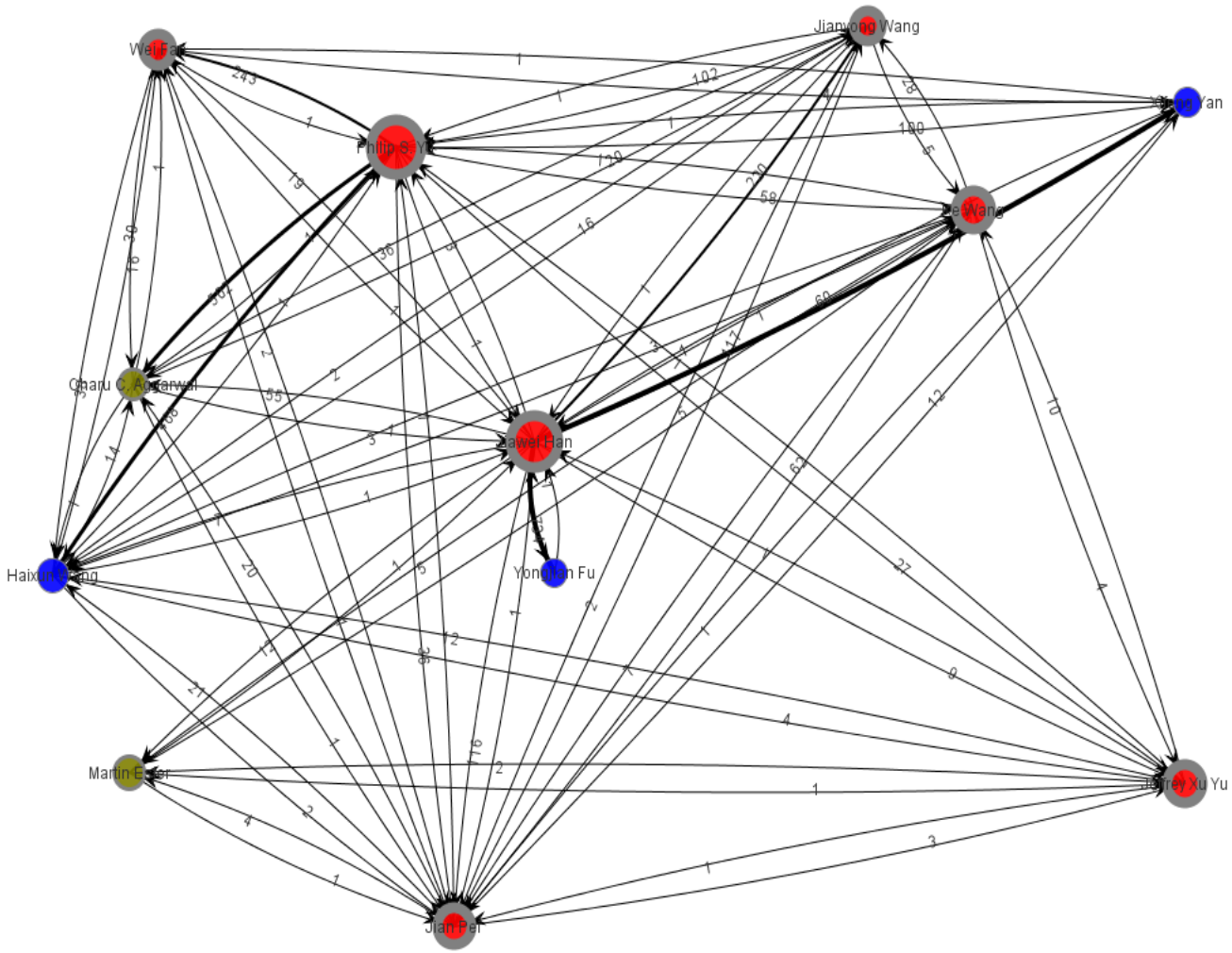
Speedup vs. #Computer nodes



Influential nodes on different topics

Dataset	Topic	Representative Nodes
Author	Data Mining	Heikki Mannila, Philip S. Yu, Dimitrios Gunopulos, Jiawei Han, Christos Faloutsos, Bing Liu, Vipin Kumar, Tom M. Mitchell, Wei Wang, Qiang Yang, Xindong Wu, Jeffrey Xu Yu, Osmar R. Zaiane
	Machine Learning	Pat Langley, Alex Waibel, Trevor Darrell, C. Lee Giles, Terrence J. Sejnowski, Samy Bengio, Daphne Koller, Luc De Raedt, Vasant Honavar, Floriana Esposito, Bernhard Scholkopf
	Database System	Gerhard Weikum, John Mylopoulos, Michael Stonebraker, Barbara Pernici, Philip S. Yu, Sharad Mehrotra, Wei Sun, V. S. Subrahmanian, Alejandro P. Buchmann, Kian-Lee Tan, Jiawei Han
	Information Retrieval	Gerard Salton, W. Bruce Croft, Ricardo A. Baeza-Yates, James Allan, Yi Zhang, Mounia Lalmas, Zheng Chen, Ophir Frieder, Alan F. Smeaton, Rong Jin
	Web Services	Yan Wang, Liang-jie Zhang, Schahram Dustdar, Jian Yang, Fabio Casati, Wei Xu, Zakaria Maamar, Ying Li, Xin Zhang, Boualem Benatallah, Boualem Benatallah
	Semantic Web	Wolfgang Nejdl, Daniel Schwabe, Steffen Staab, Mark A. Musen, Andrew Tomkins, Juliana Freire, Carole A. Goble, James A. Hendler, Rudi Studer, Enrico Motta
	Bayesian Network	Daphne Koller, Paul R. Cohen, Floriana Esposito, Henri Prade, Michael I. Jordan, Didier Dubois, David Heckerman, Philippe Smets
Citation	Data Mining	Fast Algorithms for Mining Association Rules in Large Databases, Using Segmented Right-Deep Trees for the Execution of Pipelined Hash Joins, Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data, Discovery of Multiple-Level Association Rules from Large Databases, Interleaving a Join Sequence with Semijoins in Distributed Query Processing
	Machine Learning	Object Recognition with Gradient-Based Learning, Correctness of Local Probability Propagation in Graphical Models with Loops, A Learning Theorem for Networks at Detailed Stochastic Equilibrium, The Power of Amnesia: Learning Probabilistic Automata with Variable Memory Length, A Unifying Review of Linear Gaussian Models
	Database System	Mediators in the Architecture of Future Information Systems, Database Techniques for the World-Wide Web: A Survey, The R*-Tree: An Efficient and Robust Access Method for Points and Rectangles, Fast Algorithms for Mining Association Rules in Large Databases
	Web Services	The Web Service Modeling Framework WSMF, Interval Timed Coloured Petri Nets and their Analysis, The design and implementation of real-time schedulers in RED-linux, The Self-Serv Environment for Web Services Composition
	Web Mining	Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data, Fast Algorithms for Mining Association Rules in Large Databases, The OO-Binary Relationship Model: A Truly Object Oriented Conceptual Model, Distributions of Surfers' Paths Through the World Wide Web: Empirical Characterizations, Improving Fault Tolerance and Supporting Partial Writes in Structured Coterie Protocols for Replicated Objects
	Semantic Web	FaCT and iFaCT, The GRAIL concept modelling language for medical terminology, Semantic Integration of Semistructured and Structured Data Sources, Description of the RACER System and its Applications, DL-Lite: Practical Reasoning for Rich Dls

Social Influence Sub-graph on “Data mining”



Application—Expert Finding

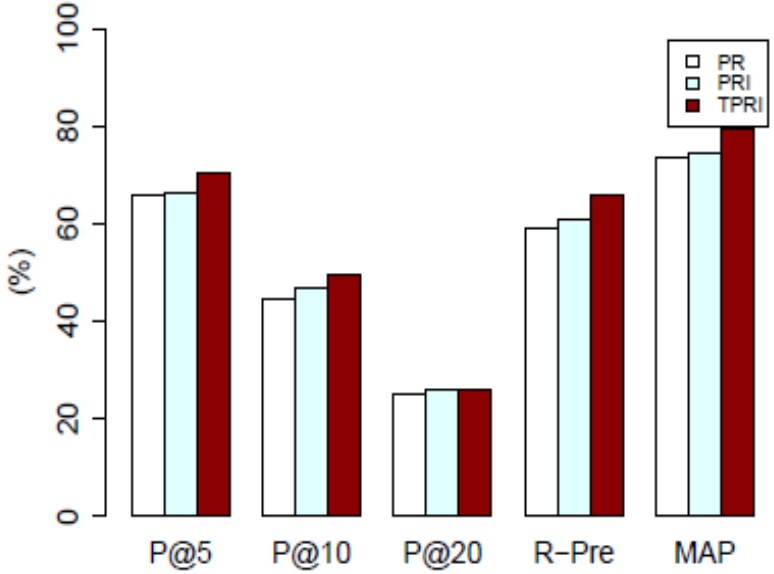


Table 7: Performance of expert finding with different approaches.

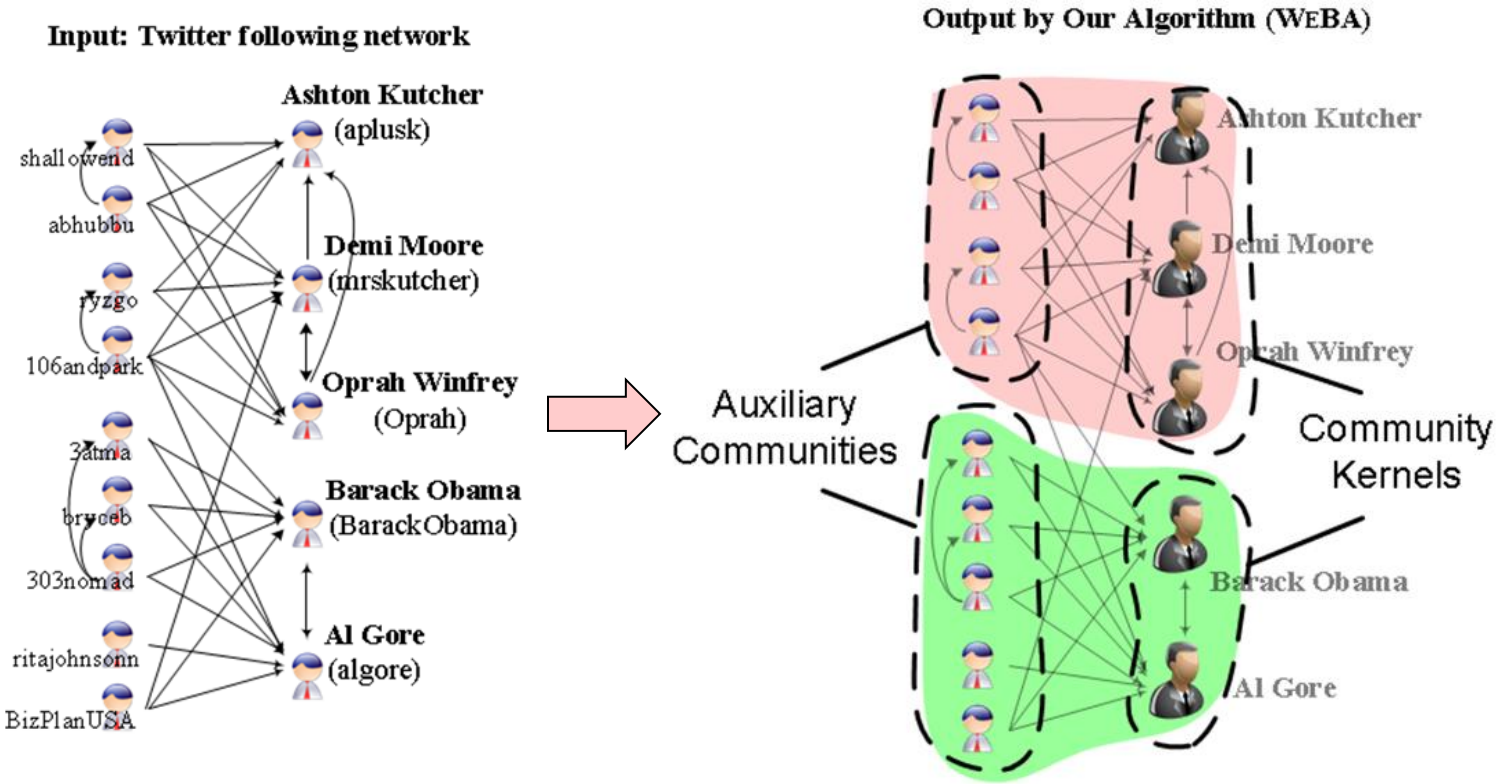
Expert finding data from (Tang, KDD08; ICDM08)

http://arnetminer.org/lab-datasets/expertfinding/

Community Kernels

- Liaoruo Wang, Tiancheng Lou, Jie Tang, and John E. Hopcroft. Detecting Community Kernels in Large Social Networks. In **ICDM 2011**. pp. 784-793.

Community Kernel

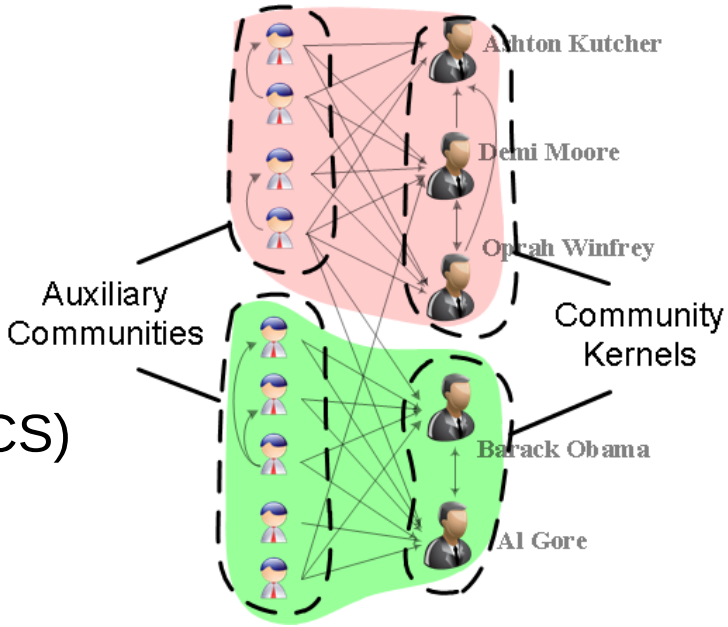


Pareto Principle: Less than 1% of the Twitter users (e.g. entertainers, politicians, writers) produce 50% of its content, while the others (e.g. fans, followers, readers) have much less influence and completely different social behavior.

Kernel users and ordinary users exhibit very different behaviors.

Approach (Greedy & WeBA)

- Challenges
  - How to identify kernel members, and
  - How to determine the structural of community kernels.
- Formalize the problem into an optimization problem.
  - Proposed two new algorithms
  - Greedy
    - Maximum cardinality search(MCS)
    - Runs in linear time
  - Global Relaxation
    - Random walk(Annealing)
    - Theoretical error bound
    - Almost linear time.

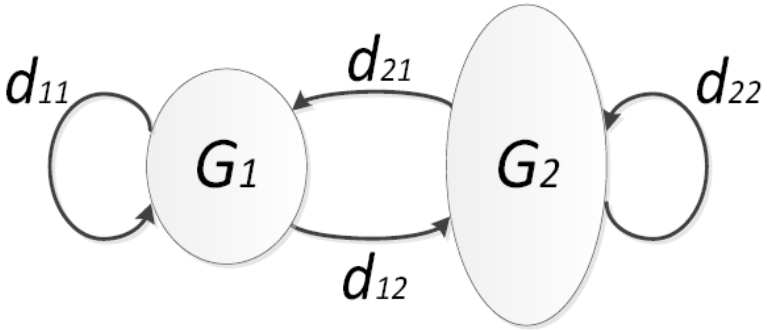


Unbalanced Weakly-Bipartite Structure

- Empirical property of many real-world networks:

$$d_{21} > d_{11} > d_{22} \gg d_{12}$$

$$d_{ij} = \frac{|E(V_i, V_j)|}{|V_j|}, \quad i, j \in \{1, 2\}$$

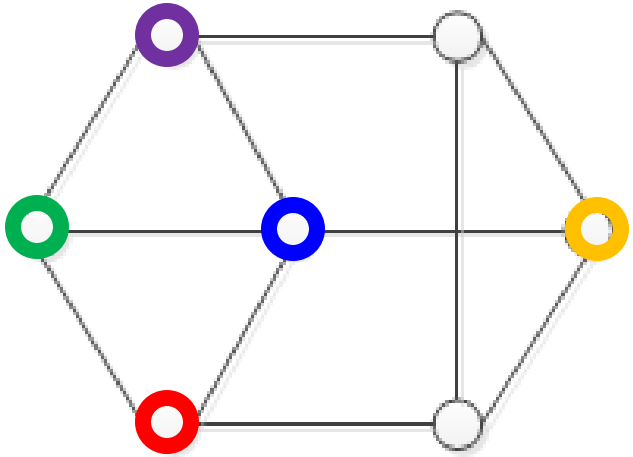


Network	d_{21}	d_{11}	d_{22}	d_{12}
Coauthor	14.19	5.34	4.42	0.37
Wikipedia	1689.31	104.22	4.69	0.60
Twitter	110.78	26.78	2.94	0.29
Slashdot	180.90	84.56	10.75	0.64
Citation	76.69	35.81	23.80	0.26

Greedy Algorithm

- Given an graph $G = (V, E)$ and a kernel size k
  - Initialize the set S to be a random vertex $v \in V$
  - Iteratively add to S the vertex with the most connections to S
  - Always pick the vertex with the highest degree

- Example



Greedy Algorithm

- Given an graph $G = (V, E)$ and a kernel size k
  - Initialize the set S to be a random vertex $v \in V$
  - Iteratively add to S the vertex with the most connections to S
  - Always pick the vertex with the highest degree

- Running time and space complexity: $O(|V| + |E|)$
- No guaranteed error bound
- Repeat $O(|V|/k)$ times to obtain steady state and reduce the effect of random selection of the initial point

WEIGHT-BALANCED ALGORITHM (WEBA)

- Each vertex $v \in V$ has a weight vector $\vec{w}(v) = \{w_1(v), \cdots, w_l(v)\}$ to represent its relative importance for each community kernel
- Optimization Problem:

$$\max \quad \mathcal{L}(\vec{w}) = \sum_{(u,v)\in E} \vec{w}(u) \cdot \vec{w}(v)$$

subject to

$$\sum_{v\in V} w_i(v) = k, \ \forall i \in \{1, \cdots, l\}$$
$$\sum_{1\le i\le l} w_i(v) \le 1, \ \forall v \in V$$
$$w_i(v) \ge 0, \forall v \in V, \ \forall i \in \{1, \cdots, l\}$$

- Intractable to solve — we approximate the solution by iteratively solving its one-dimensional version $\mathcal{L}(w)$

WEIGHT-BALANCED ALGORITHM (WEBA)

- Theorem 1: A global maximum of the objective function $\mathcal{L}(w)$ corresponds to a community kernel.
- Given an graph $G = (V, E)$ and a kernel size k, maximizing $\mathcal{L}(w)$ is NP-hard.
  - Initialize the set S to be a random subset obtained by GREEDY
  - Assign weight 1 to each vertex in S and weight 0 otherwise
  - If $\exists u, v \in V$ such that $w(u) < 1, w(v) > 0$ and $nw(u) > nw(v)$, where $nw(u)$ is the neighboring weight of u, the weights of u and v are modified to locally maximize $\mathcal{L}(w)$

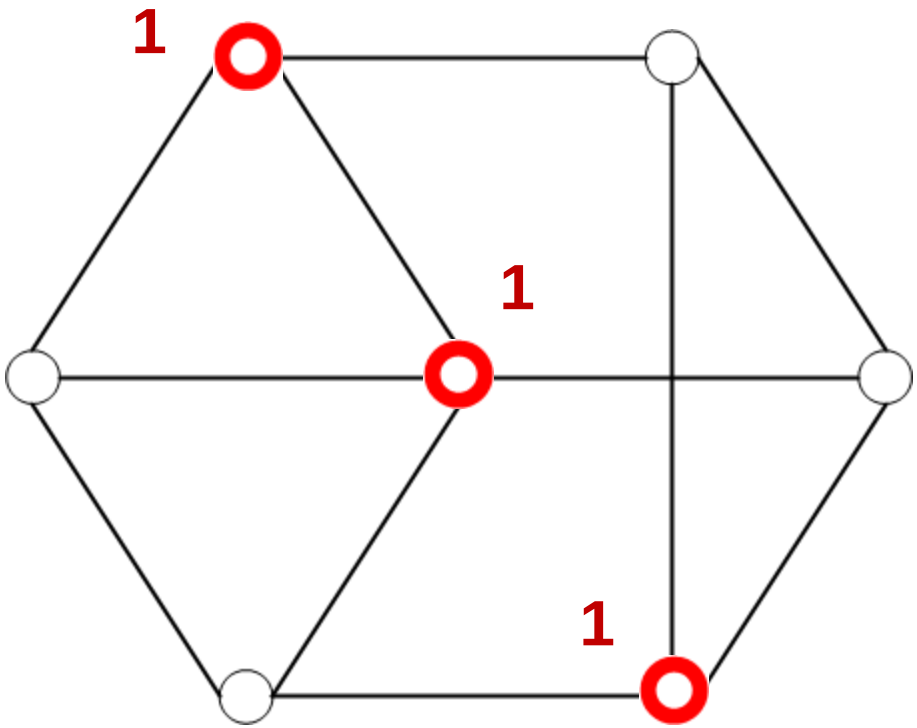
relaxation conditions

WEBA

Input: $G = (V, E)$ and kernel size k
Output: community kernels $\mathbf{K} = \{\mathcal{K}_1, \mathcal{K}_2, \cdots, \mathcal{K}_\ell\}$
$\mathbf{K} \leftarrow \emptyset$
repeat
  $S \leftarrow$ a subset returned by $\textsc{Greedy}(G, k)$
  $\forall v \in S,\ w(v) \leftarrow 1;\ \forall v \notin S,\ w(v) \leftarrow 0$
  while $\exists\ u, v \in V$ *satisfying the relaxation conditions* **do**
    if $(u, v) \notin E$ **then** $\delta \leftarrow \min\{1 - w(u), w(v)\}$
    else $\delta \leftarrow \min\left\{1 - w(u), w(v), \dfrac{nw(u) - nw(v)}{2}\right\}$
    pick one pair $\{u, v\}$ with the maximum δ value
    $w(u) \leftarrow w(u) + \delta,\ \ w(v) \leftarrow w(v) - \delta$
  $C \leftarrow \{v \in V \mid w(v) = 1\}$
  if $C \notin \mathbf{K}$ **then** $\mathbf{K} \leftarrow \{\mathbf{K}, C\}$
until $O(|V|/k)$ *times*;
return $\mathbf{K}$

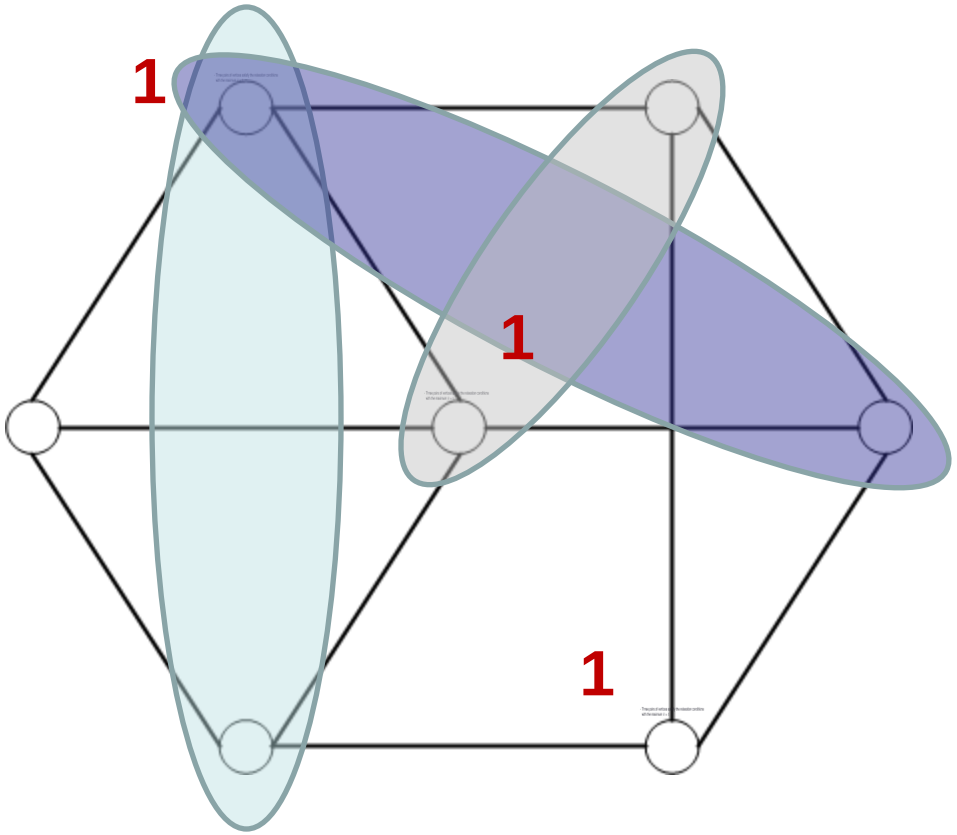
An Example

- Given a graph and a kernel size $k = 3$
- Given a random subset of size k



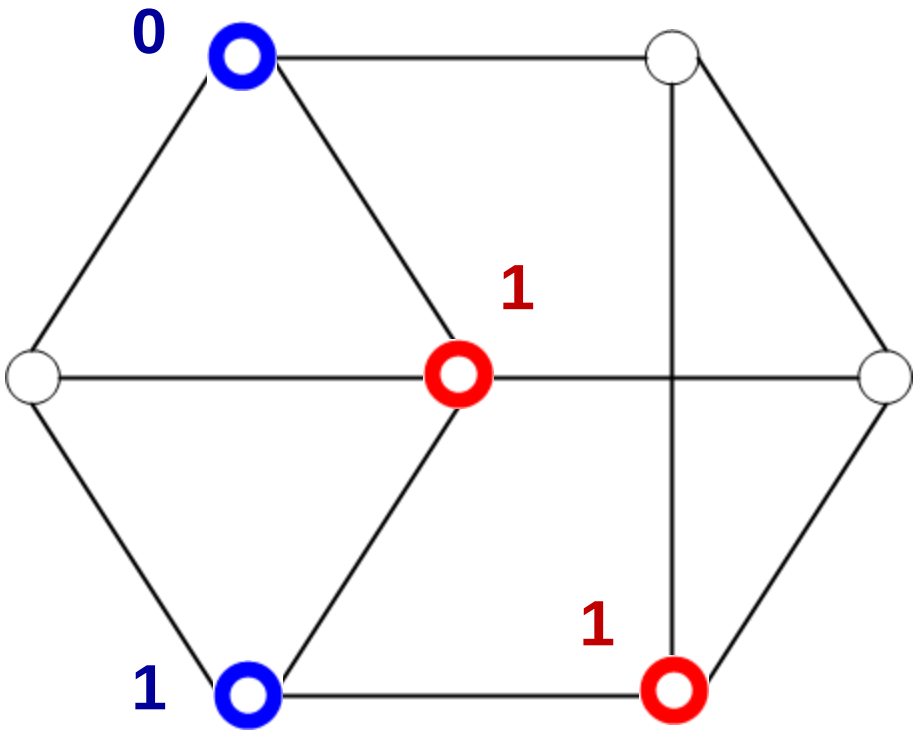
Weight-Balanced Algorithm (WeBA)

- Three pairs of vertices satisfy the relaxation conditions with the maximum $\delta = 1$



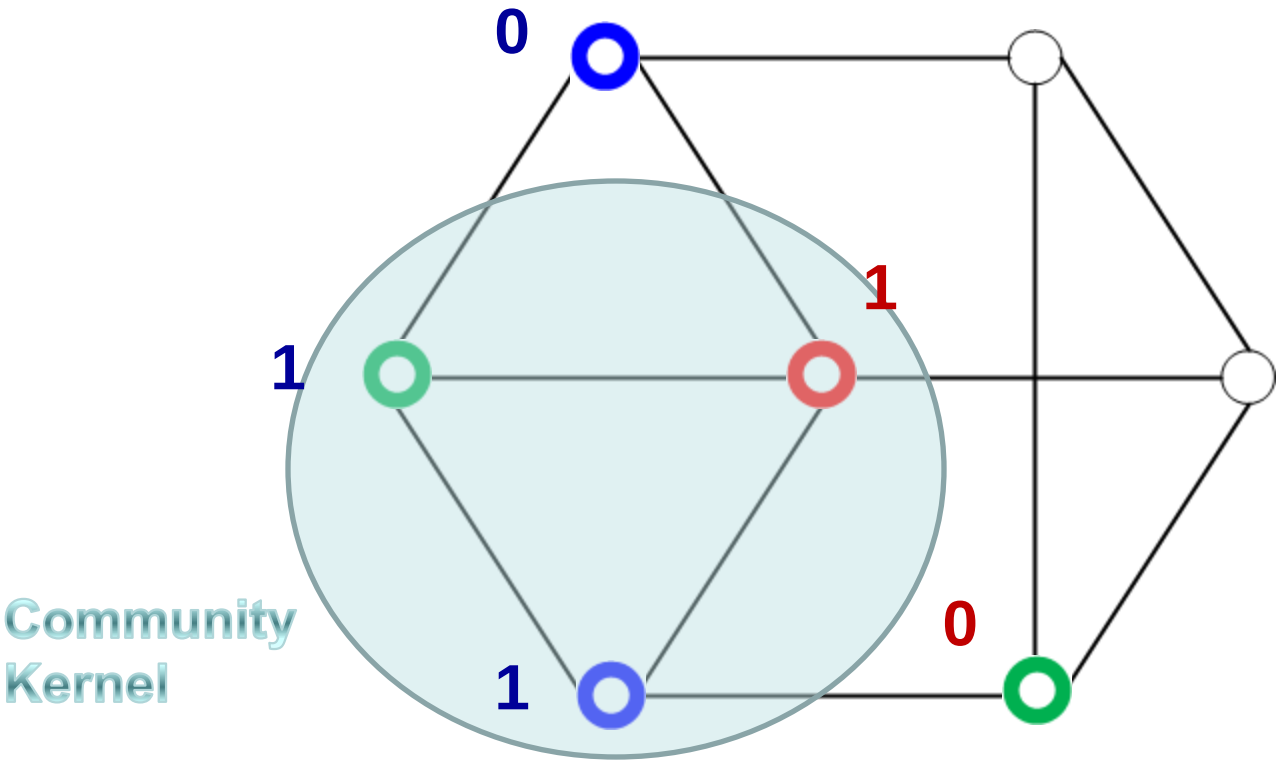
WEIGHT-BALANCED ALGORITHM (WEBA)

- $w(u) \leftarrow w(u) + \delta \Longrightarrow w(u) \leftarrow 1$
- $w(v) \leftarrow w(v) - \delta \Longrightarrow w(v) \leftarrow 0$



WEIGHT-BALANCED ALGORITHM (WEBA)

- Keep balancing weights as described above until no pairs of vertices satisfy the relaxation conditions



WEBA

- Theorem 2 (correctness):

  WEBA is guaranteed to converge to a feasible solution.

- Theorem 3 (error bound):

  For any assigned weights $\{w(v), \forall v \in V\}$ and any $\varepsilon > 0$, after

$$\max\left\{\frac{4k^3D^5}{\varepsilon^2}, \frac{2mkD^3}{\varepsilon}\right\}$$

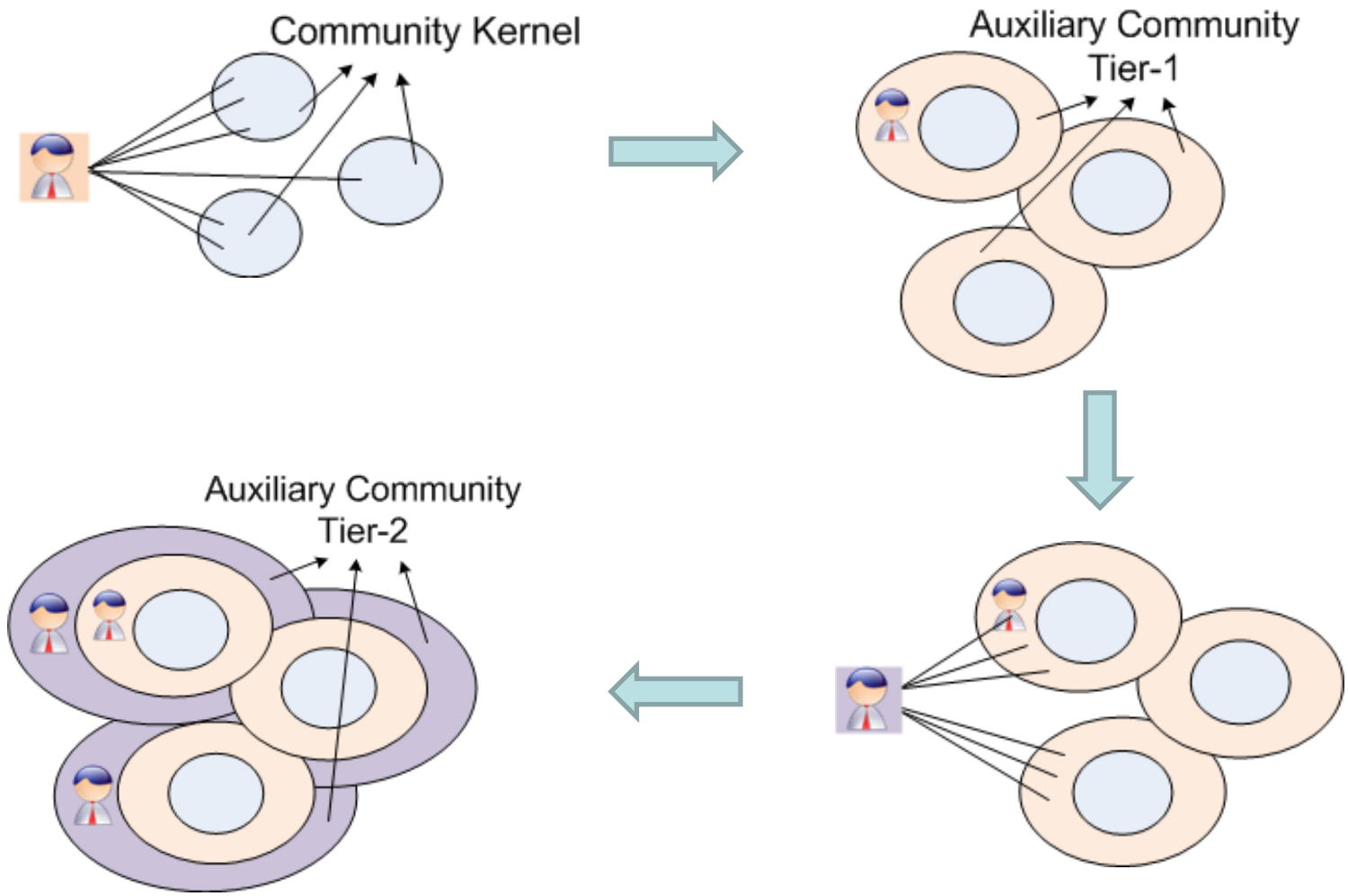
  iterations, we have $\mathcal{L}(w^*(v)) - \mathcal{L}\big(w(v)\big) \leq \varepsilon$.

- Repeat $O(|V|/k)$ times to obtain steady state and reduce the effect of random selection of the initial point

Finding Auxiliary Community

- Given community kernels $\{K_1, K_2, \cdots, K_l\}$
  - Label each vertex that is not in any kernel as unassociated
  - For each unassociated vertex, rank the kernels according to the number of edges from the vertex to each kernel and the vertices that have already been associated with that kernel
  - Associate the vertex with the top-ranked kernel(s)
  - Repeat this process until no more vertices can be associated
- Auxiliary communities can overlap with each other

Finding Auxiliary Community

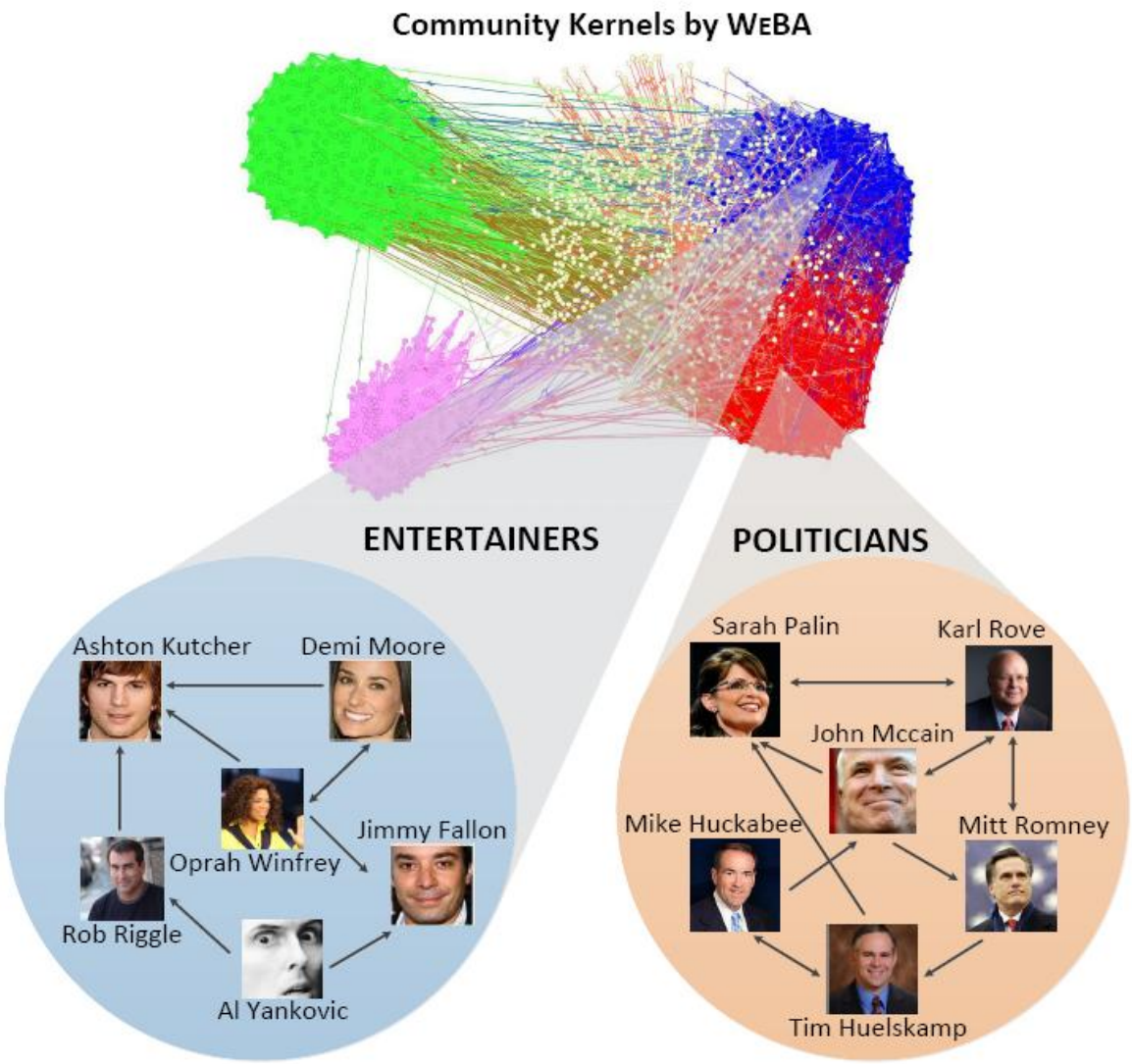


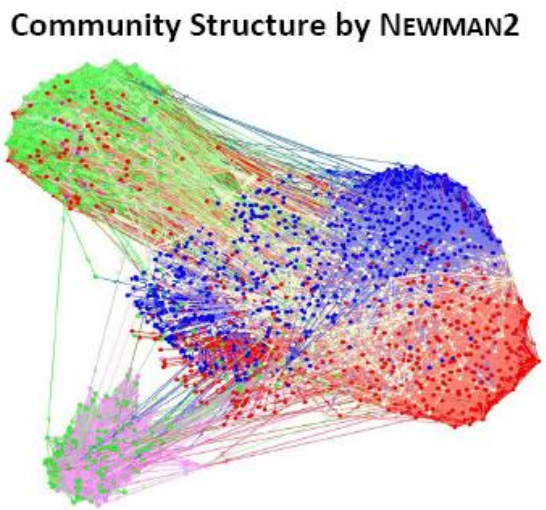
Experimental Results

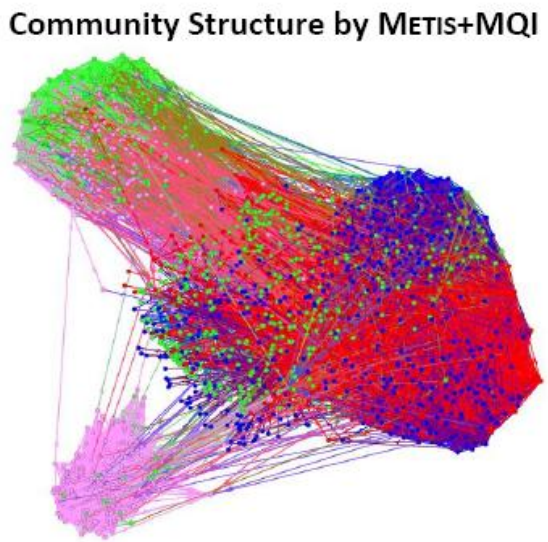
- Data Sets
  - Coauthor (822,415 nodes; 2,928,360 edges)
    - Benchmark coauthor network (52,146 nodes; 134,539 edges)
  - Wikipedia (310,990 nodes; 10,780,996 edges)
    - Namespace talk pages (263 nodes; 1,075 edges)
    - User personal pages (266 nodes; 33,829 edges)
  - Twitter (465,023 nodes; 833,590 edges)
- Algorithms

Local Spectral Partitioning (LSP)	METIS+MQI
d-LSP (high-degree)	NEWMAN1 (betweenness)
p-LSP (high-PageRank)	NEWMAN2 (modularity)
α-β	LOUVAIN

Case Study on Twitter







Results on Coauthor & Wikipedia

- **On average, WEBA improves Precision by 340% (wiki) and 70% (coauthor), and improves Recall by 130% (wiki) and 41% (coauthor).**

	Precision						Recall					
	wiki		coauthor				wiki		coauthor			
	Talk	User	AI	…	NC	Average	Talk	User	AI	…	NC	Average
LSP	0.061	0.085	0.502	…	0.342	0.573	0.171	0.315	0.458	…	0.398	0.561
d-LSP	0.051	0.091	0.528	…	0.504	0.617	0.427	0.273	0.519	…	0.463	0.609
p-LSP	0.046	0.082	0.678	…	0.403	0.641	0.442	0.237	0.337	…	0.491	0.574
METIS+MQI	0.049	0.012	0.847	…	0.055	0.488	0.062	0.361	0.089	…	0.077	0.379
LOUVAIN	0.063	0.122	0.216	…	0.272	0.437	0.388	0.348	0.184	…	0.19	0.343
NEWMAN1	0.033	0.203	0.4	…	0.259	0.431	0.069	0.070	0.306	…	0.174	0.311
NEWMAN2	0.039	0.085	0.298	…	0.613	0.463	0.029	0.075	0.364	…	0.467	0.335
α-β	0.324	0.336	0.443	…	0.747	0.626	0.422	0.427	**0.602**	…	0.568	0.654
WEBA	**0.456**	**0.46**	**0.852**	…	**0.837**	**0.911**	**0.589**	**0.57**	0.577	…	**0.582**	**0.664**
GREEDY	0.334	0.403	0.83	…	0.746	0.752	0.432	0.499	0.545	…	0.56	0.659

87%

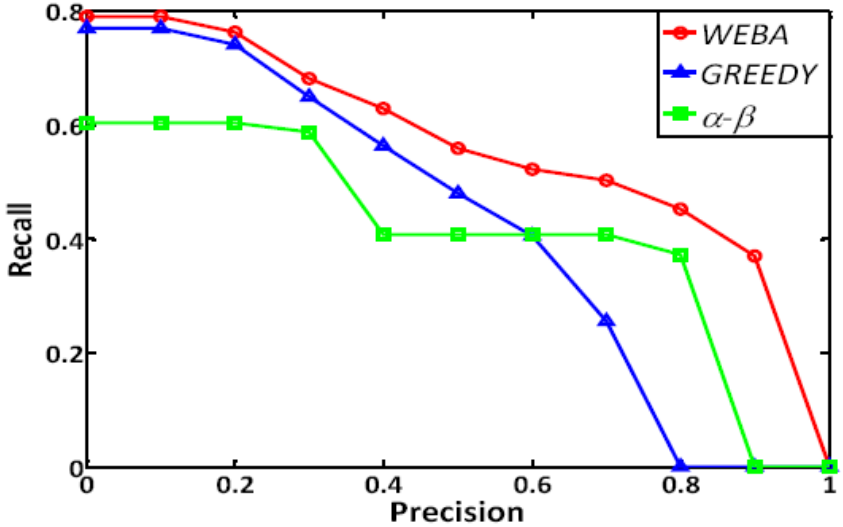
Results on Coauthor & Wikipedia

- **On average, WEBA increases F1-score by 300% (wiki) and 61% (coauthor), and increases Resemblance by 180% (wiki) and 67% (coauthor).**

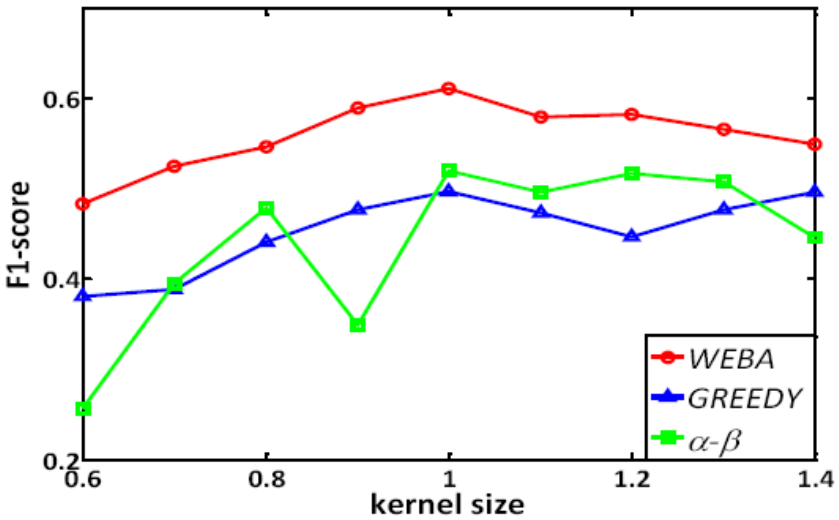
	F1-score						Resemblance (Jaccard Index)					
	wiki		coauthor				wiki		coauthor			
	Talk	User	AI	…	NC	Average	Talk	User	AI	…	NC	Average
LSP	0.090	0.134	0.479	…	0.368	0.565	0.177	0.175	0.143	…	0.138	0.169
d-LSP	0.091	0.137	0.524	…	0.483	0.612	0.175	0.149	0.164	…	0.204	0.193
p-LSP	0.083	0.121	0.450	…	0.443	0.595	0.177	0.153	0.130	…	0.208	0.194
METIS+MQI	0.055	0.023	0.162	…	0.064	0.370	0.130	0.090	0.022	…	0.018	0.048
LOUVAIN	0.108	0.181	0.199	…	0.224	0.361	0.212	0.245	0.101	…	0.102	0.118
NEWMAN1	0.014	0.111	0.346	…	0.208	0.347	0.127	0.208	0.139	…	0.119	0.120
NEWMAN2	0.033	0.080	0.327	…	0.53	0.350	0.131	0.148	0.137	…	0.198	0.130
α-β	0.367	0.376	0.510	…	0.646	0.587	0.436	0.444	0.178	…	0.227	0.203
WEBA	**0.514**	**0.509**	**0.688**	…	**0.686**	**0.763**	**0.561**	**0.557**	**0.234**	…	**0.259**	**0.246**
GREEDY	0.377	0.446	0.658	…	0.64	0.696	0.445	0.503	0.216	…	0.234	0.222

30%

SENSITIVITY



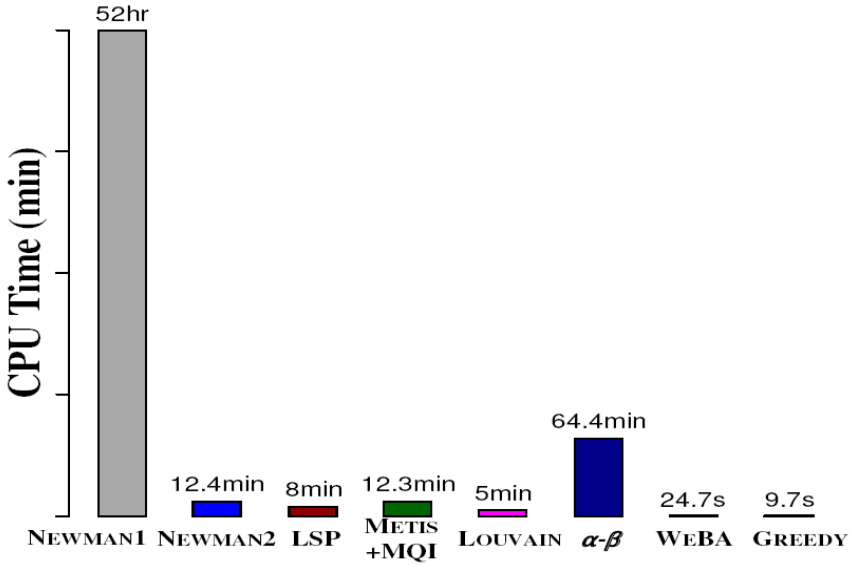
(a) Precision vs. Recall



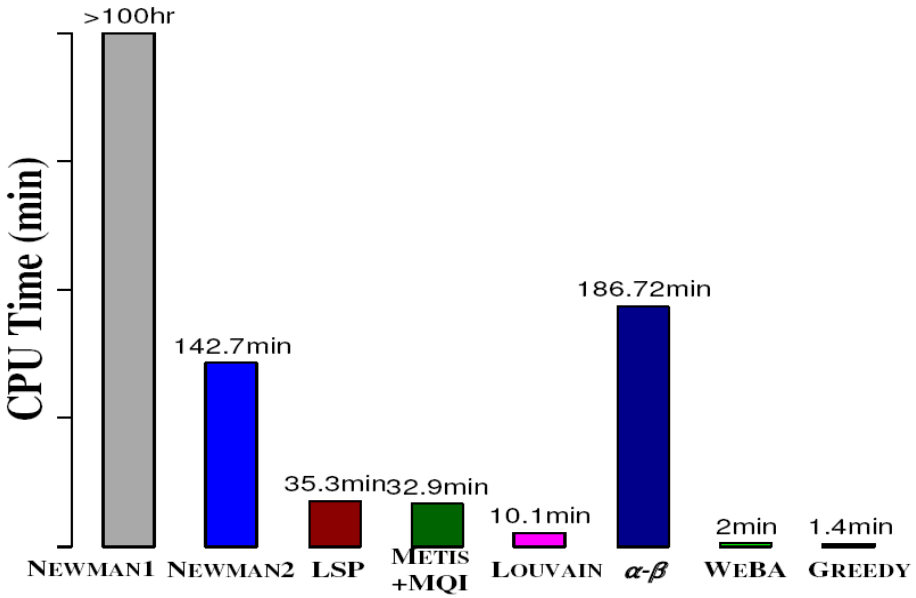
(b) F1-score vs. kernel size

EFFICIENCY — TWITTER & COAUTHOR

465,023 nodes, 833,590 edges



822,415 nodes, 2,928,360 edges

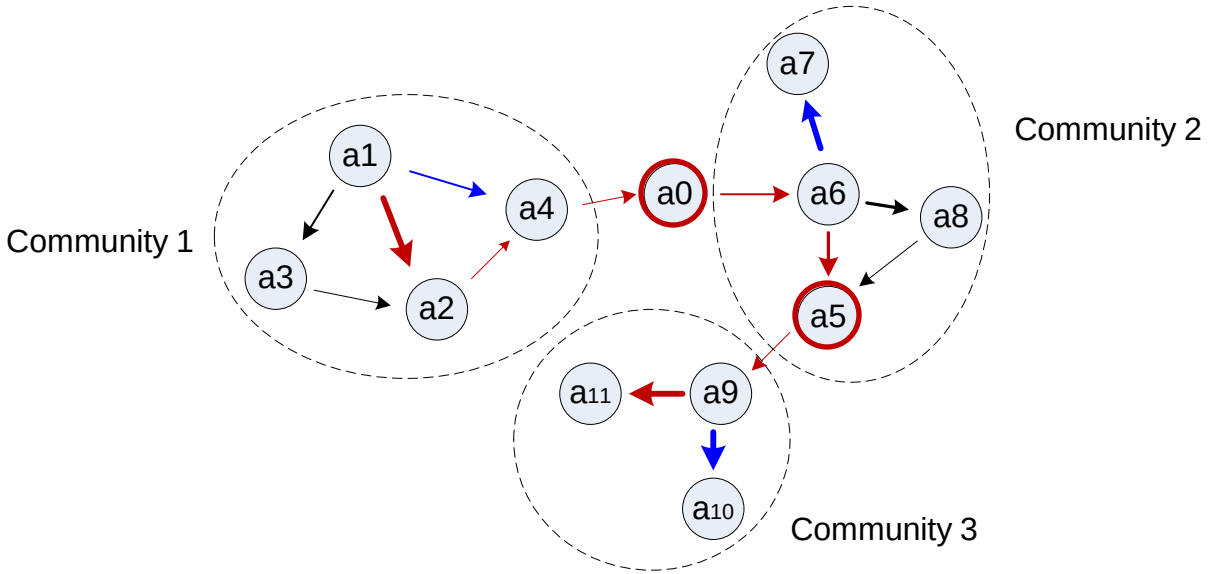


Top-*k* Structural Holes

-Tiancheng Lou and Jie Tang. Mining Top-k Structural Holes in Large Networks. (Submitted)

Structural Hole Detection

- The **structural hole** theory suggests that:
  - Individuals would benefit from filling the “holes” **between communities** that are otherwise disconnected.
  - Structural holes play a key role in the **information diffusion**.
- Lack of a principled methodology to discover structural holes from a given social network.



Problem Definition

- Top-k Structural Holes Detection.
  - Denote $G = (V, E)$ as a social network, with l communities $\mathbf{C} = (C_1, C_2, \ldots, C_l)$
  - Denote top-k structural holes V_{SH} as a subset of k nodes, which maximize the following qualify function:

    $$\max Q(V_{SH}, \mathbf{C}), \text{ with } |V_{SH}| = k$$

  - Since the utility function Q is a general definition, which can be instantiated in different ways,
- Our contribution
  - We develop two instantiation models based on the above objective function.

Model 1 : HIS

- Opinion leader vs. Structural holes

$$I(v,i) = \max_{(u,v)\in E, i\in S} \{I(v,i), \alpha_i I(u,i) + \beta_S H(u,S)\}$$

$$H(v,S) = \min_{i\in S} \{I(v,i)\}$$

- Analyze
  - Condition of existence
  - ε- convergence property
  - Iterative algorithm and improved algorithm
  - Parameter analysis

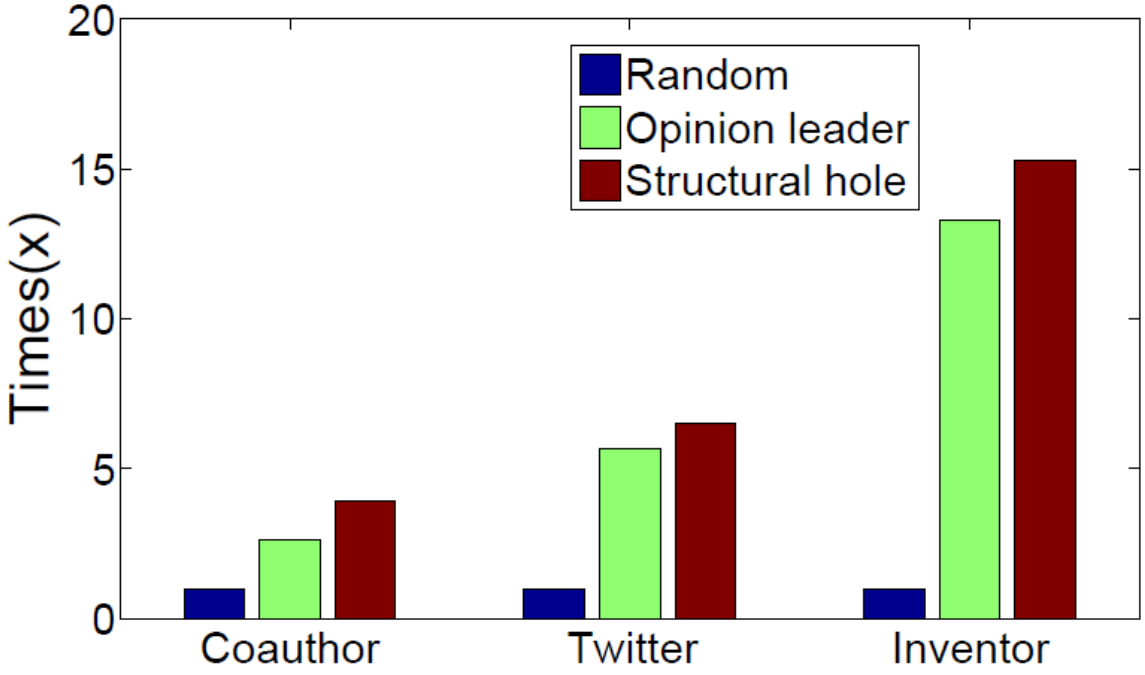
Model 2 : MaxD

- Define $Q(V_{SH}, C) = MC(G, C) – MC(G \setminus V_{SH}, C)$
  - Where $MC(G, C)$ is the minimal cut of communities C in G.
- Analyze
  - NP-Hardness of the exact solution, the first attempt to proof minimal node-cut problem in an un-weighted graph.
  - Suppose the k-DENSEST SUBGRAPH is hard to approximate within $n^{\Omega(1)}$, then the problem is also hard to approximate within $n^{\Omega(1)}$ as well.
  - Approximation algorithm and evaluation methods.

Data Sets

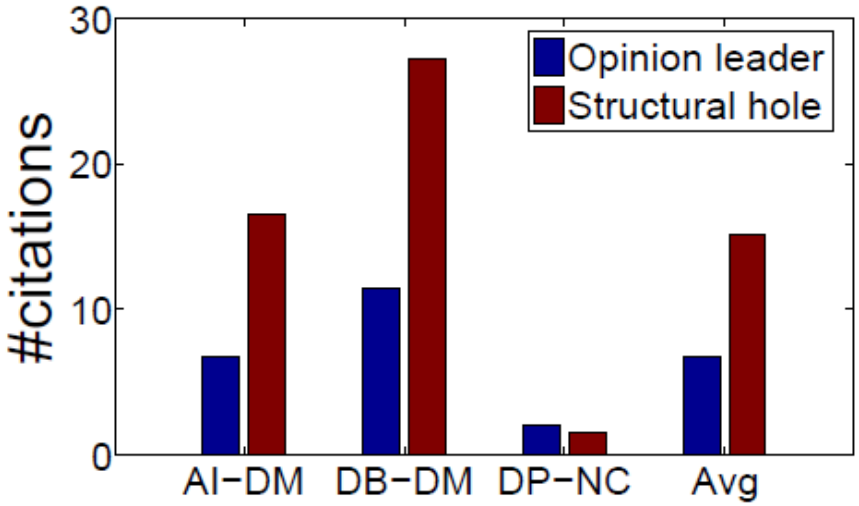
- **Coauthors:** 815,946 authors and 2,792,833 coauthorships
  - Six areas: Artificial Intelligence (AI), Databases (DB), Data Mining (DM), Distributed Parallel Computing (DP), Graphics, Vision and HCI (GV), as well as Networks, Communications and Performance (NC)
  - View PC members in more than one areas as spanning structural holes
- **Twitter:** 13,442,659 users and 56,893,234 links
  - Communities are discovered by the community kernel detection algorithm
- **Inventor:** 2,445,351 inventors and 5,841,940 co-inventing relationships
  - Each company is a community

Observation 1

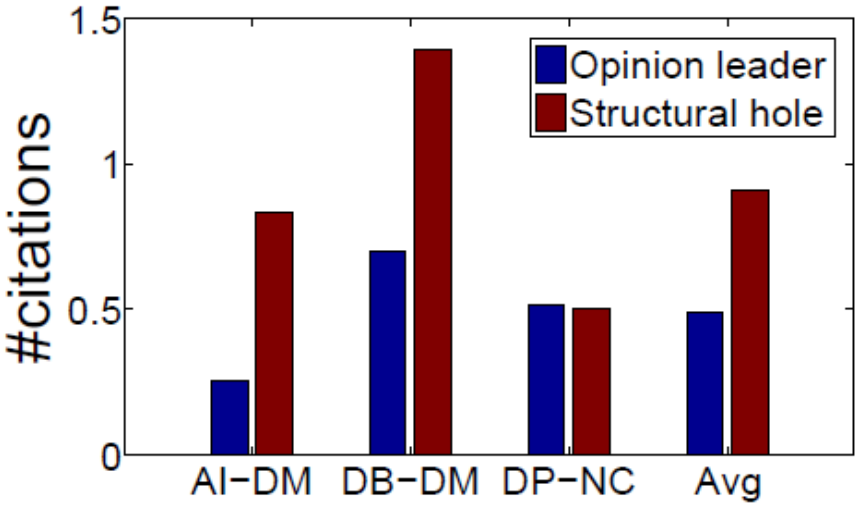


Structural hole nodes are more likely to connect importance nodes than opinion leaders

Observation 2



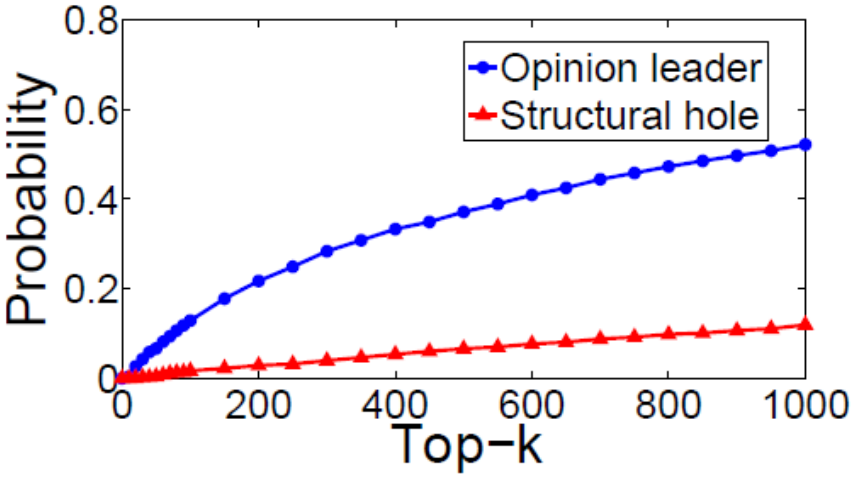
(a) Cross domain



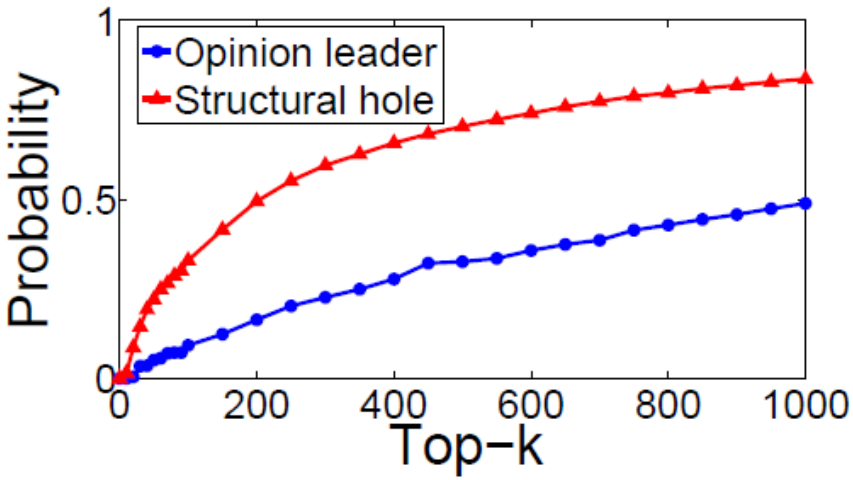
(b) Outer domain

Structural hole nodes receive more cross-domain citations than opinion leaders.

Observation 3



(a) Inner domain

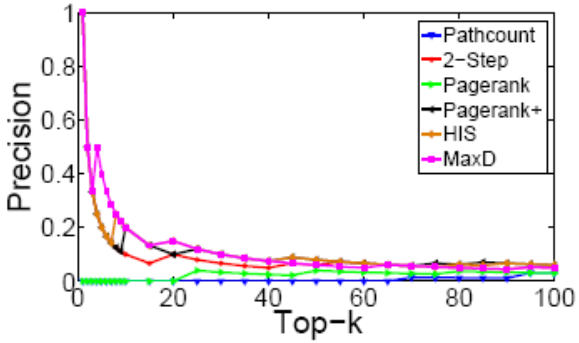


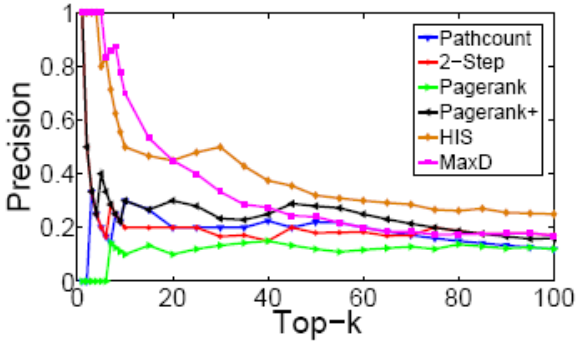
(b) Cross domain

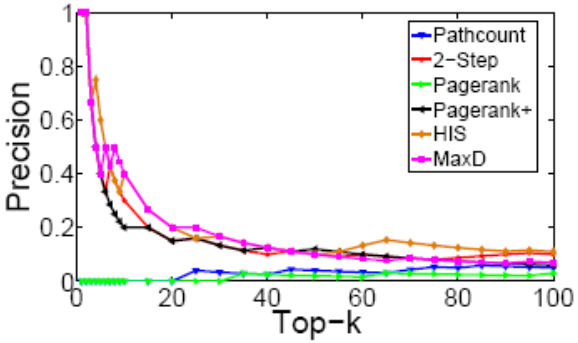
Opinion leaders play a key role in spreading information within a community, while structural hole nodes are more important for spreading information between communities

Results on Coauthor

- Use Coauthor as the benchmark dataset to evaluate the accuracy of the proposed models.
  - Both models clearly outperform the comparison algorithms by +20 – 40%.
  - Structural holes are determined by not only bridging positions, but also status of neighbors.

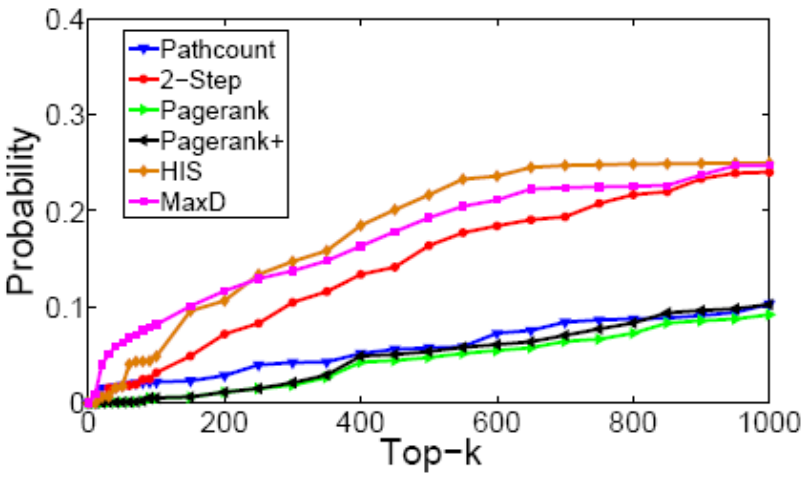


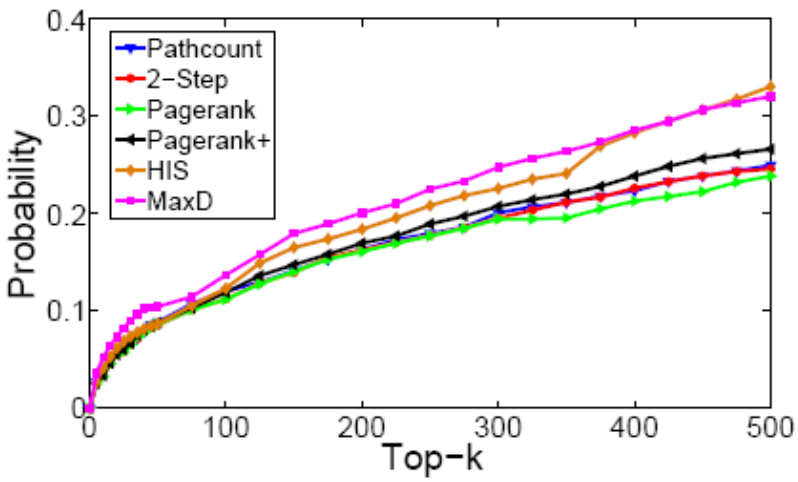




Results on Twitter

- Information spread between different communities.
  - On Twitter : top 0.2% structural hole users almost influence 10% of the tweets-forwarding behaviors between different communities.
  - On Coauthor : Improvement is statistically significant (p << 0.01)





Case Study on Inventor Network

- Most of the detected structural holes have been working in more than one job.

Table 2: Top-5 structural hole nodes discovered by our algorithms on the Inventor network. Names with ∗ are inventors with the highest (top-5) PageRank scores.

Inventor	HIS	MaxD	Title
E. Boyden		√	Professor (MIT Media Lab)
			Associate Professor (MIT McGovern Inst.)
			Group Leader (Synthetic Neurobiology)
A. Czarnik		√	Founder and Manager (Protia, LLC)
			Visiting Professor (University of Nevada)
			Co-Founder (Chief Scientific Officer)
T. Kondo*	√	√	Senior vice president (Sony Corporation)
A. Nishio		√	Director of Operations (WBI)
			Director of Department Responsible (IDA)
E. Nowak*	√		Senior vice President (Walt Disney)
			Secretary of Trustees (The New York Eye)
A. Rofougaran	√		Consultant (various wireless companies)
			Co-founder (Innovent System Corp.)
			Leader (RF-CMOS).
M. Rofougaran	√	√	Engineering Director (Broadcom Corp.)
			Co-founder(Iran Today Publications)
S. Yamazaki*	√		President and majority shareholder (SEL)

Thank you !

QA?

Data & Code:

http://arnetminer.org/lab-datasets/soinf
http://arnetminer.org/stnt