Detecting Community Kernels in Large Social Networks

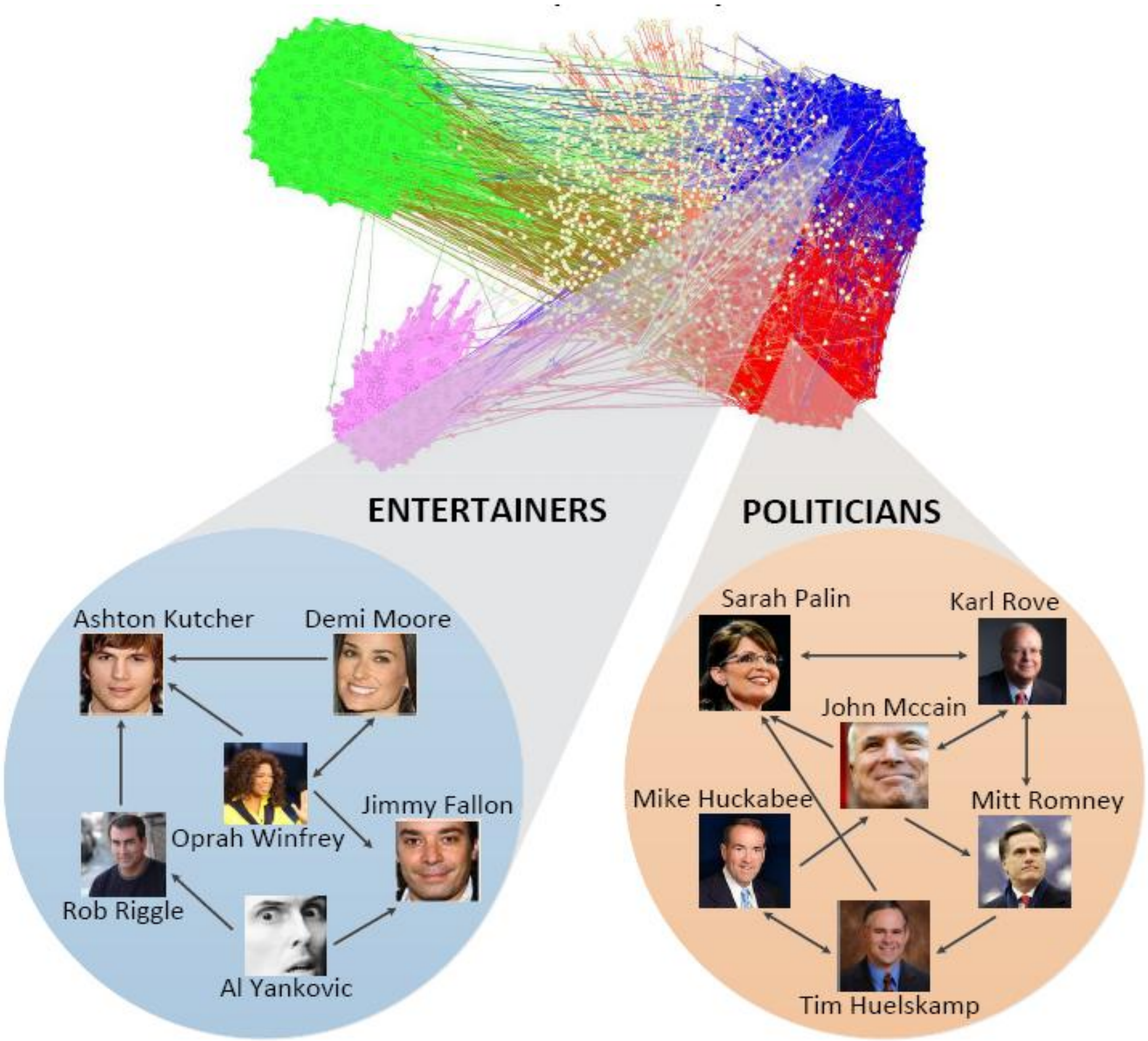
Liaoruo (Laura) Wang
Cornell University
December 14, 2011

Joint work with Tiancheng Lou, Jie Tang, and John Hopcroft

OUTLINE

- Introduction
- Problem Definition
  - Community Kernel
  - Auxiliary Community
  - Unbalanced Weakly-Bipartite Structure
- Algorithms
  - GREEDY
  - WEBA
- Experimental Results
  - Case Study
  - Quantitative Performance
  - Efficiency and Scalability

An Example

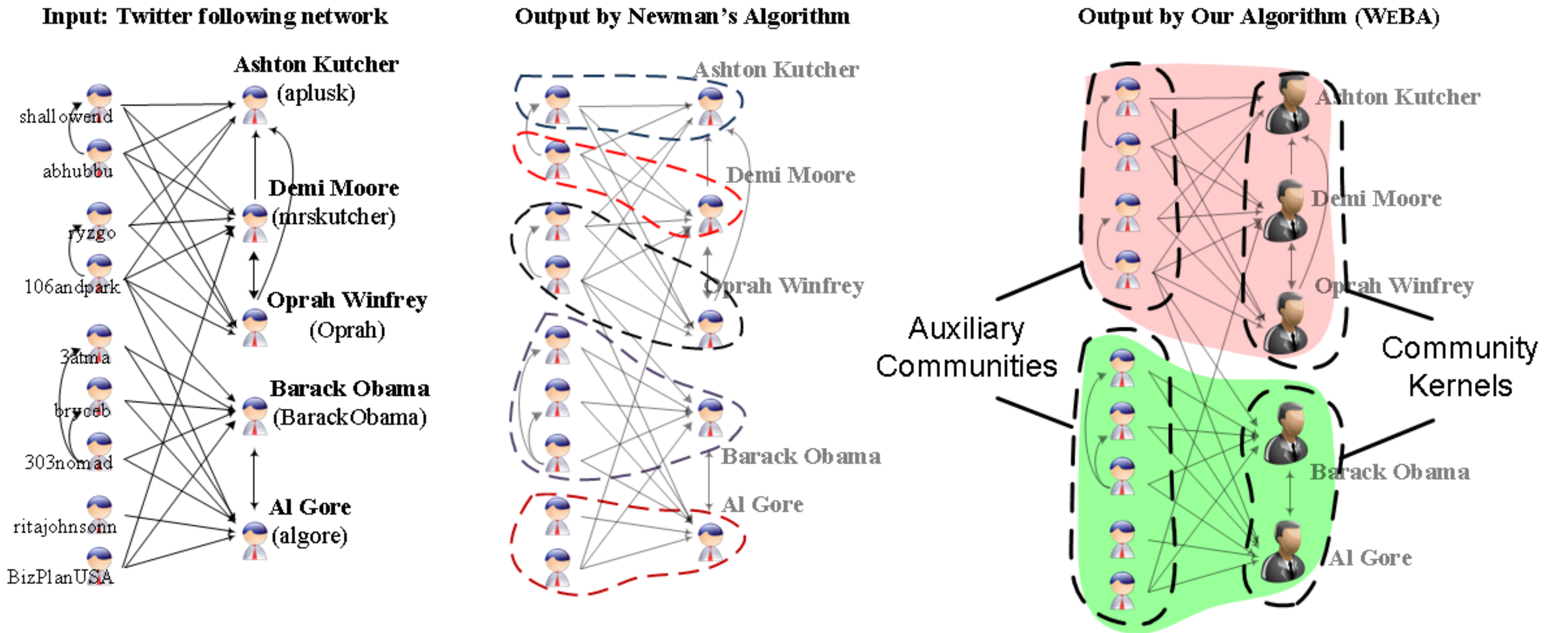


ENTERTAINERS
POLITICIANS
Ashton Kutcher
Demi Moore
Oprah Winfrey
Jimmy Fallon
Rob Riggle
Al Yankovic
Sarah Palin
Karl Rove
John Mccain
Mike Huckabee
Mitt Romney
Tim Huelskamp

Outline

- Introduction
- Problem Definition
  - Community Kernel
  - Auxiliary Community
  - Unbalanced Weakly-Bipartite Structure
- Algorithms
  - GREEDY
  - WEBA
- Experimental Results
  - Case Study
  - Quantitative Performance
  - Efficiency and Scalability

COMMUNITY KERNEL AND AUXILIARY COMMUNITY



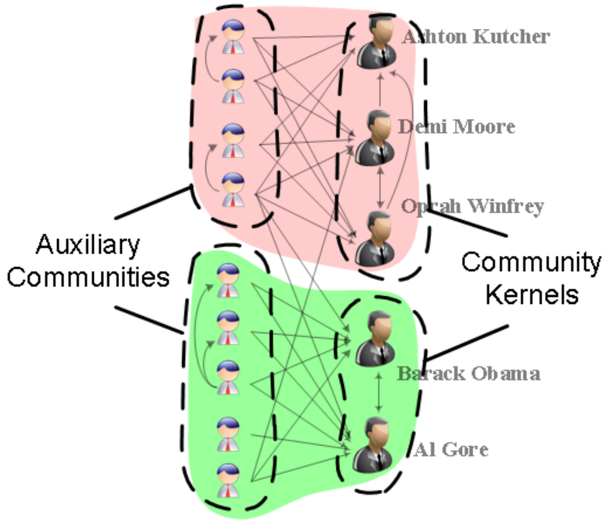
In many social networks, there exist two types of users that exhibit different influence and different behavior.

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.

DEFINITION

Given a graph $G = (V, E)$, l disjoint subsets $\{K_1, K_2, \cdots, K_l\}$ of vertices are called community kernels and l associated subsets $\{A_{k_1}, A_{k_2}, \cdots, A_{k_l}\}$ of vertices are called auxiliary communities if

- Each kernel member has more connections to/from the kernel than a vertex outside the kernel does.
- A community kernel is disjoint from its auxiliary community.
- Each auxiliary member has more connections to its associated kernel than to any other kernel.
- Each kernel member is followed by more vertices in its auxiliary community than those in the kernel.



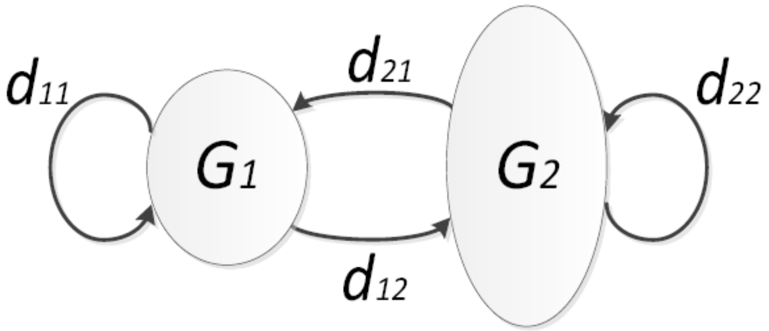
Problem: how to identify kernel members and auxiliary members, and how to determine the structure of community kernels?

Unbalanced Weakly-Bipartite (UWB) 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

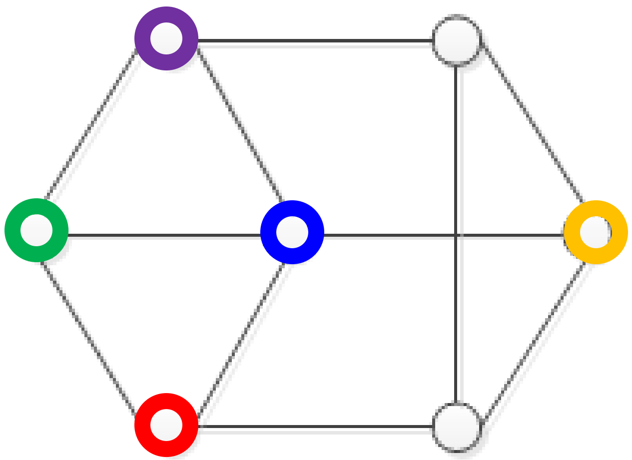
Outline

- Introduction
- Problem Definition
  - Community Kernel
  - Auxiliary Community
  - Unbalanced Weakly-Bipartite Structure
- Algorithms
  - Greedy
  - WeBA
- Experimental Results
  - Case Study
  - Quantitative Performance
  - Efficiency and Scalability

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

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

$$\textstyle\sum_{1\le i\le l} w_i(v) \le 1, \ \forall v \in V$$

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

- 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 $\mathcal{L}(w)$

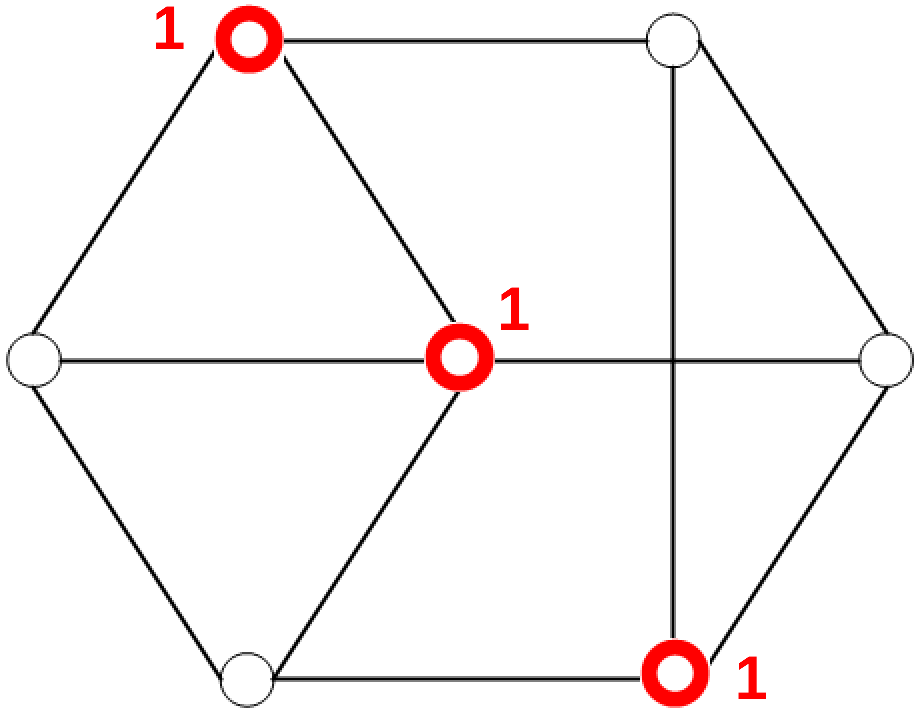
relaxation conditions

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 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 $\delta \leftarrow \min\left\{1 - w(u), w(v), \frac{nw(u) - nw(v)}{2}\right\}$
    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 $\mathbf{K}$

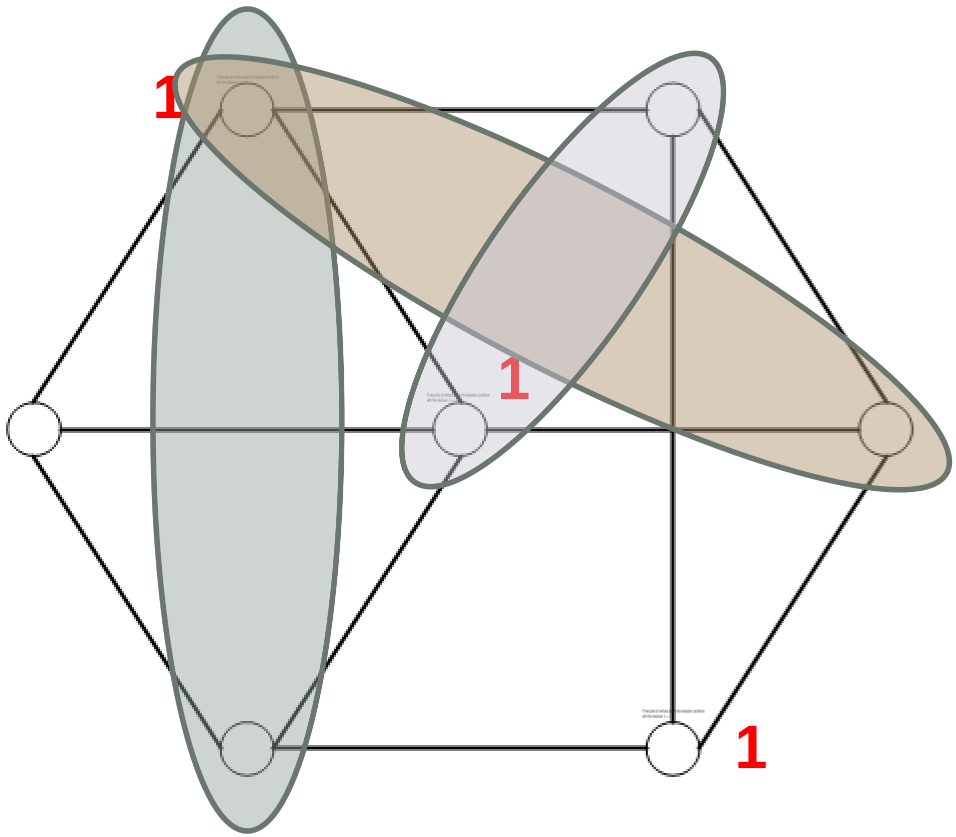
WEIGHT-BALANCED ALGORITHM (WEBA)

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



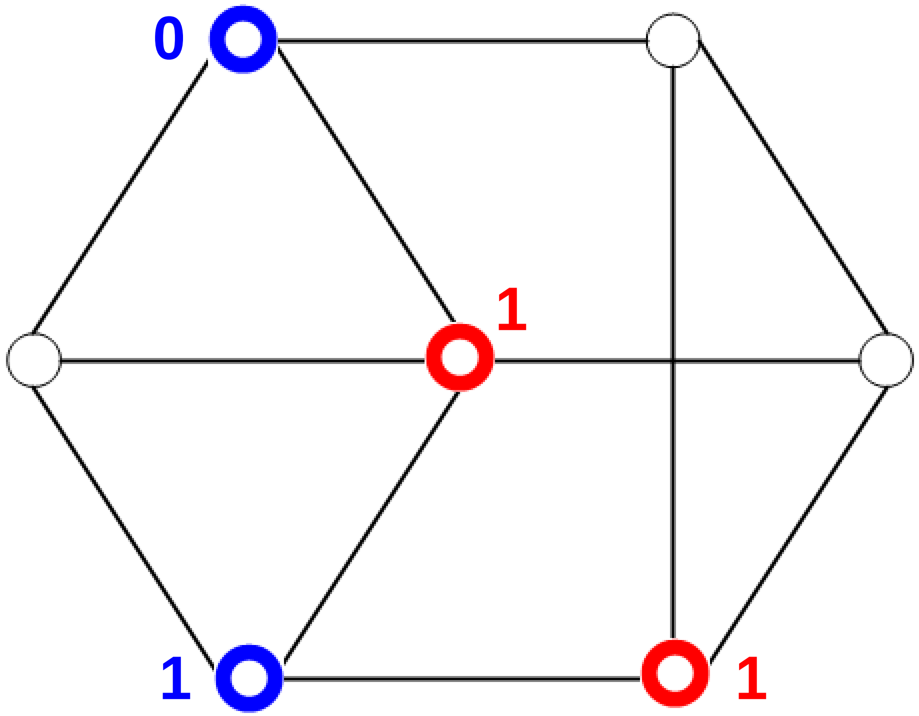
WEIGHT-BALANCED ALGORITHM (WEBA)

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



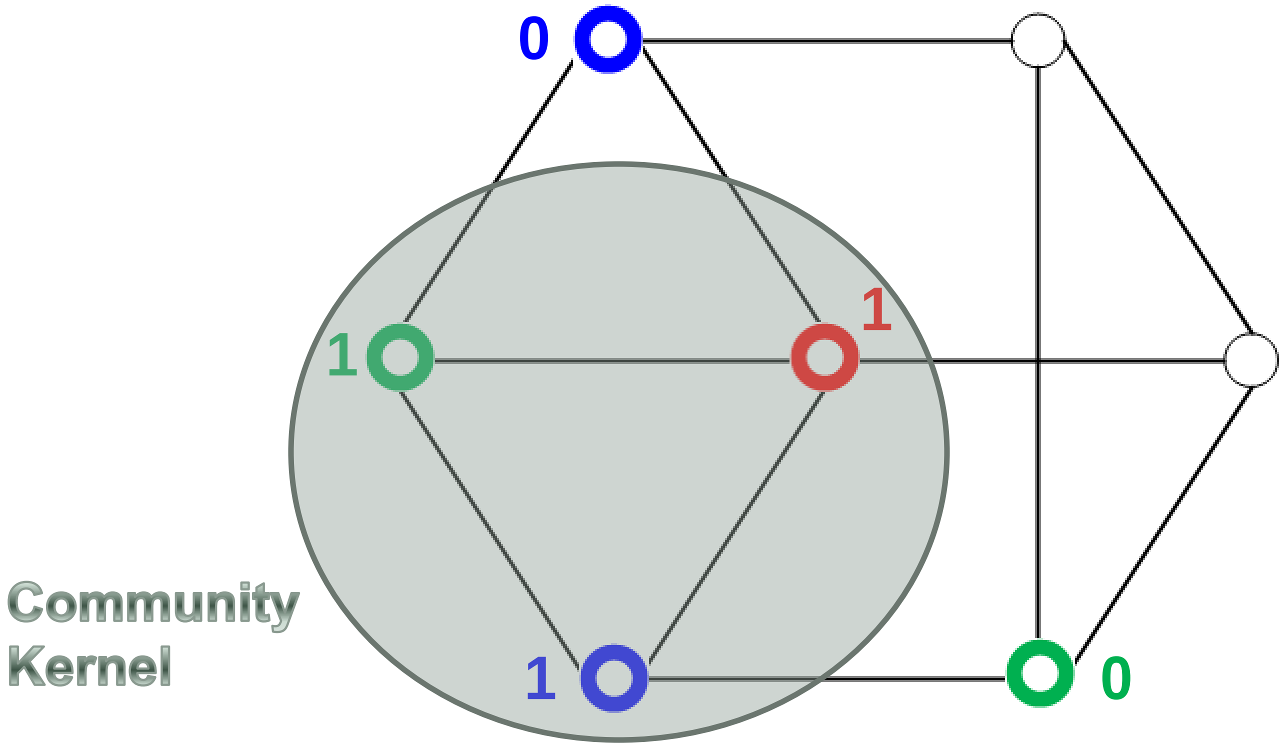
WEIGHT-BALANCED ALGORITHM (WEBA)

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



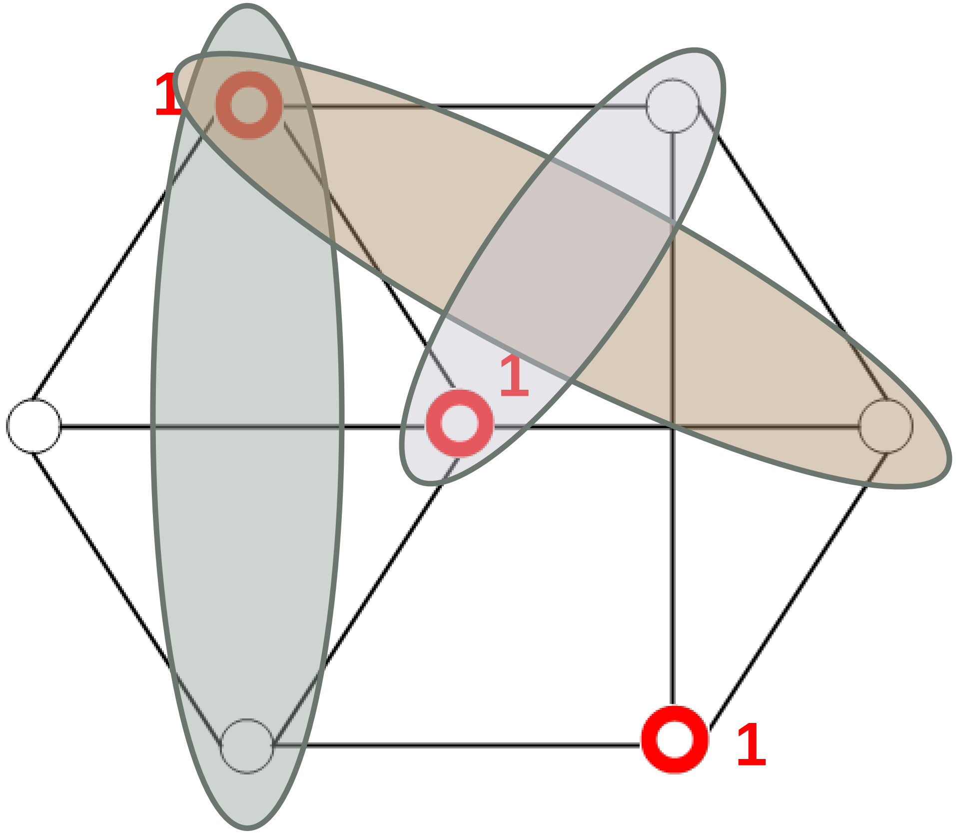
WEIGHT-BALANCED ALGORITHM (WEBA)

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



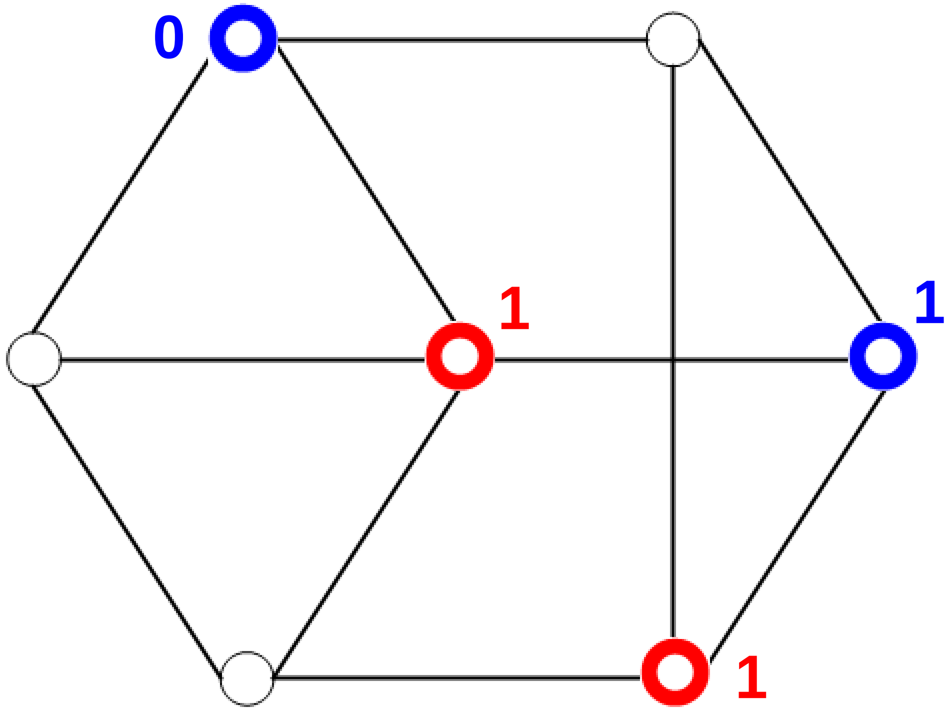
WEIGHT-BALANCED ALGORITHM (WEBA)

- Now we select another pair of vertices



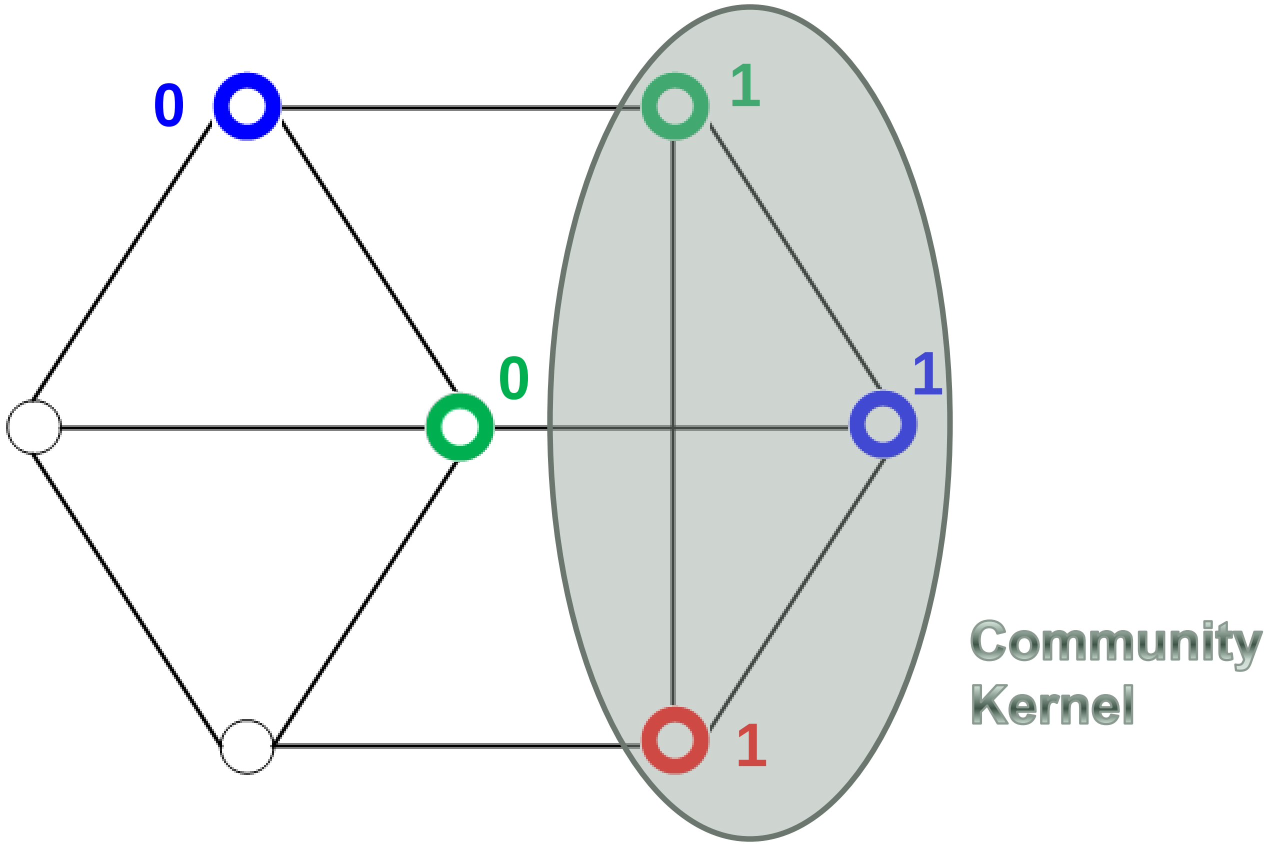
Weight-Balanced Algorithm (WeBA)

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



WEIGHT-BALANCED ALGORITHM (WEBA)

- The algorithm converges to another community kernel



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

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

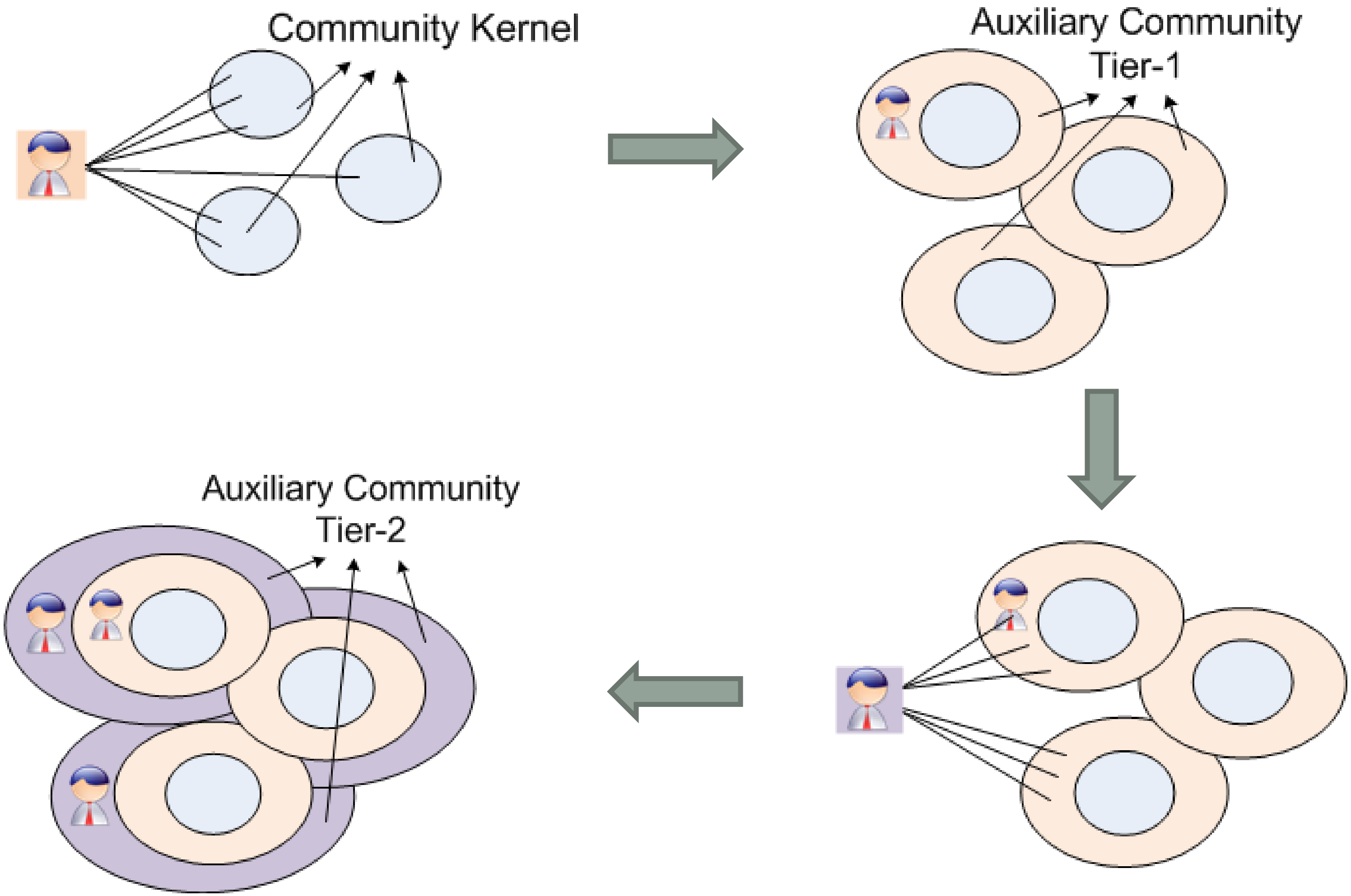
  iterations, we have $\mathcal{L}(w^*(v)) - \mathcal{L}\big(w(v)\big) \leq \varepsilon$.

- 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



Outline

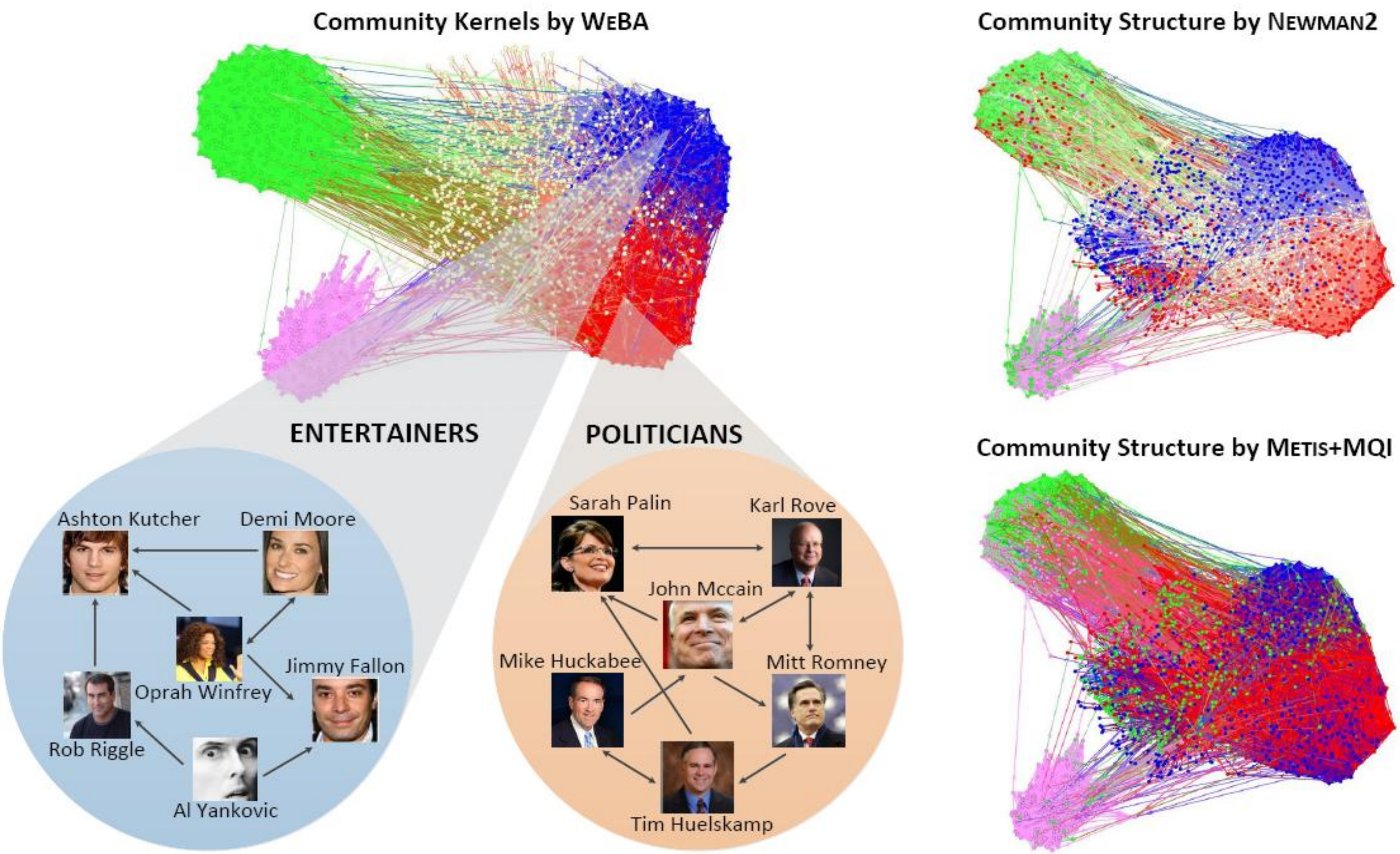
- Introduction
- Problem Definition
  - Community Kernel
  - Auxiliary Community
  - Unbalanced Weakly-Bipartite Structure
- Algorithms
  - Greedy
  - WeBA
- Experimental Results
  - Case Study
  - Quantitative Performance
  - Efficiency and Scalability

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



EXPERIMENTAL RESULTS

- 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	0.009	0.077	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%

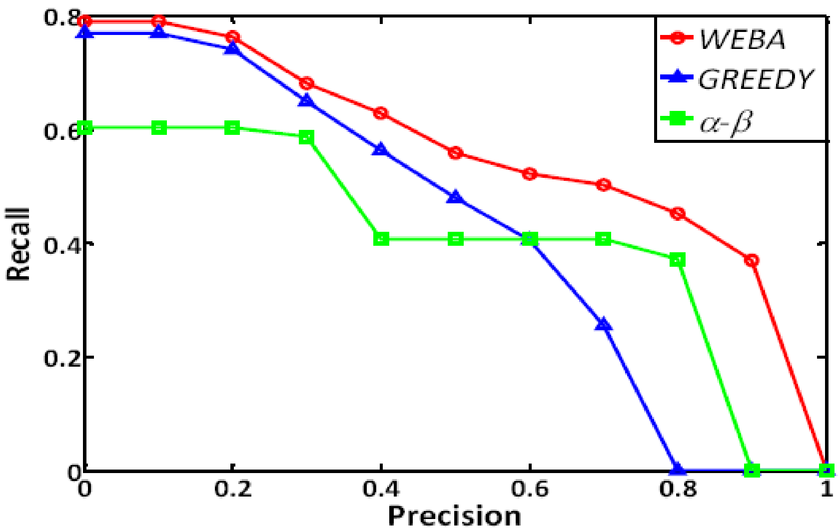
EXPERIMENTAL RESULTS

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

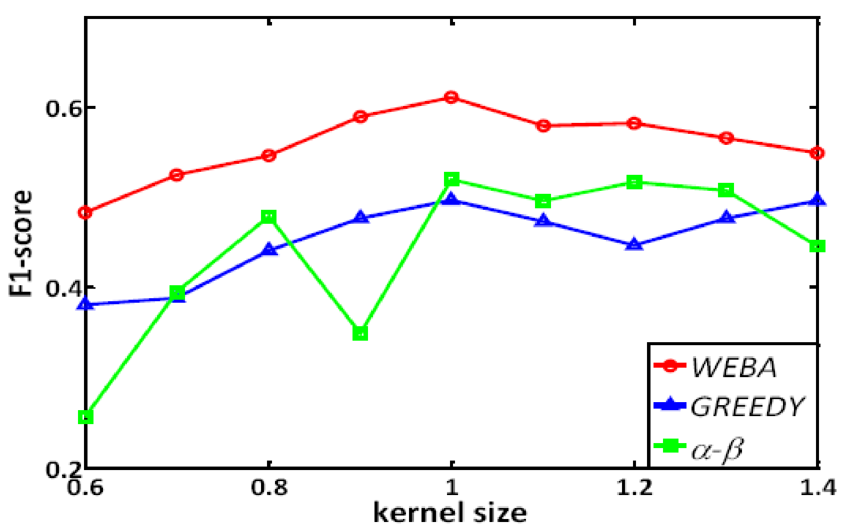
	F1-score						Resemblance (Jaccard Index)					
	wiki		coauthor				wiki		coauthor			
	Talk	User	AI	…	NC	Average	Talk	User	AI	…	NC	Average
LSP	0.090	0.134	0.479	…	0.368	0.565	0.177	0.175	0.143	…	0.138	0.169
d-LSP	0.091	0.137	0.524	…	0.483	0.612	0.175	0.149	0.164	…	0.204	0.193
p-LSP	0.083	0.121	0.450	…	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



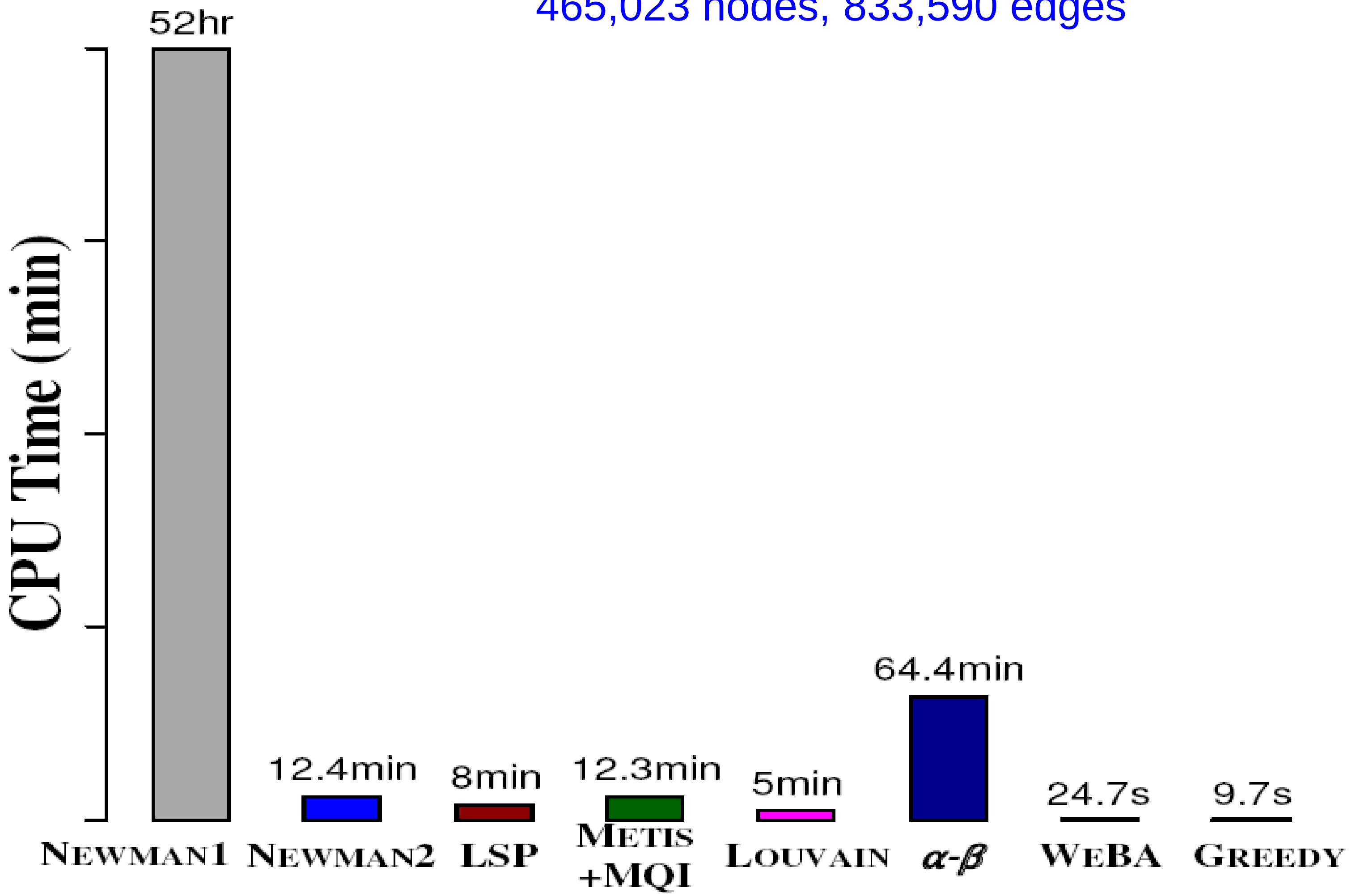
(a) Precision vs. Recall



(b) F1-score vs. kernel size

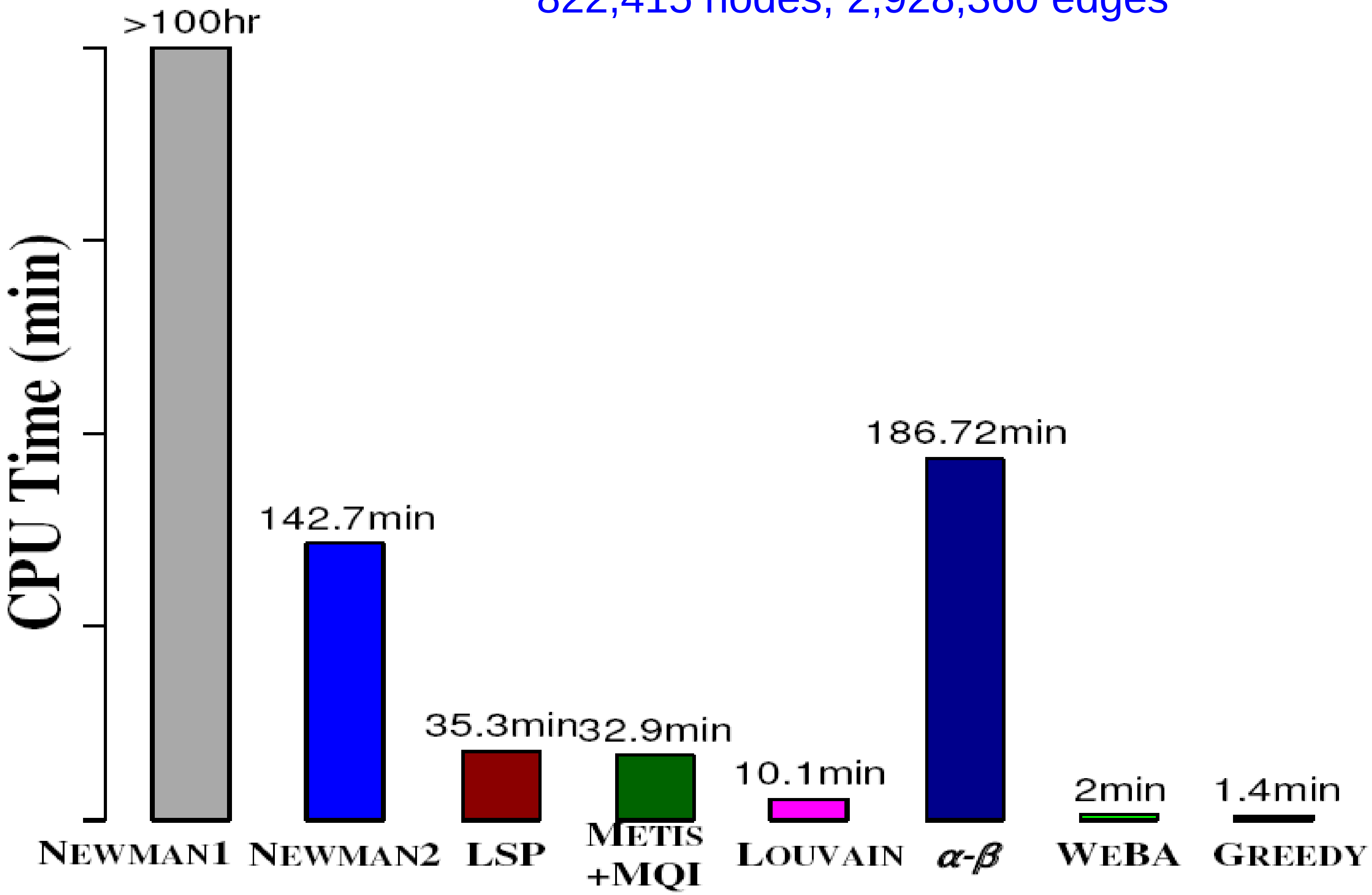
EFFICIENCY — TWITTER

465,023 nodes, 833,590 edges

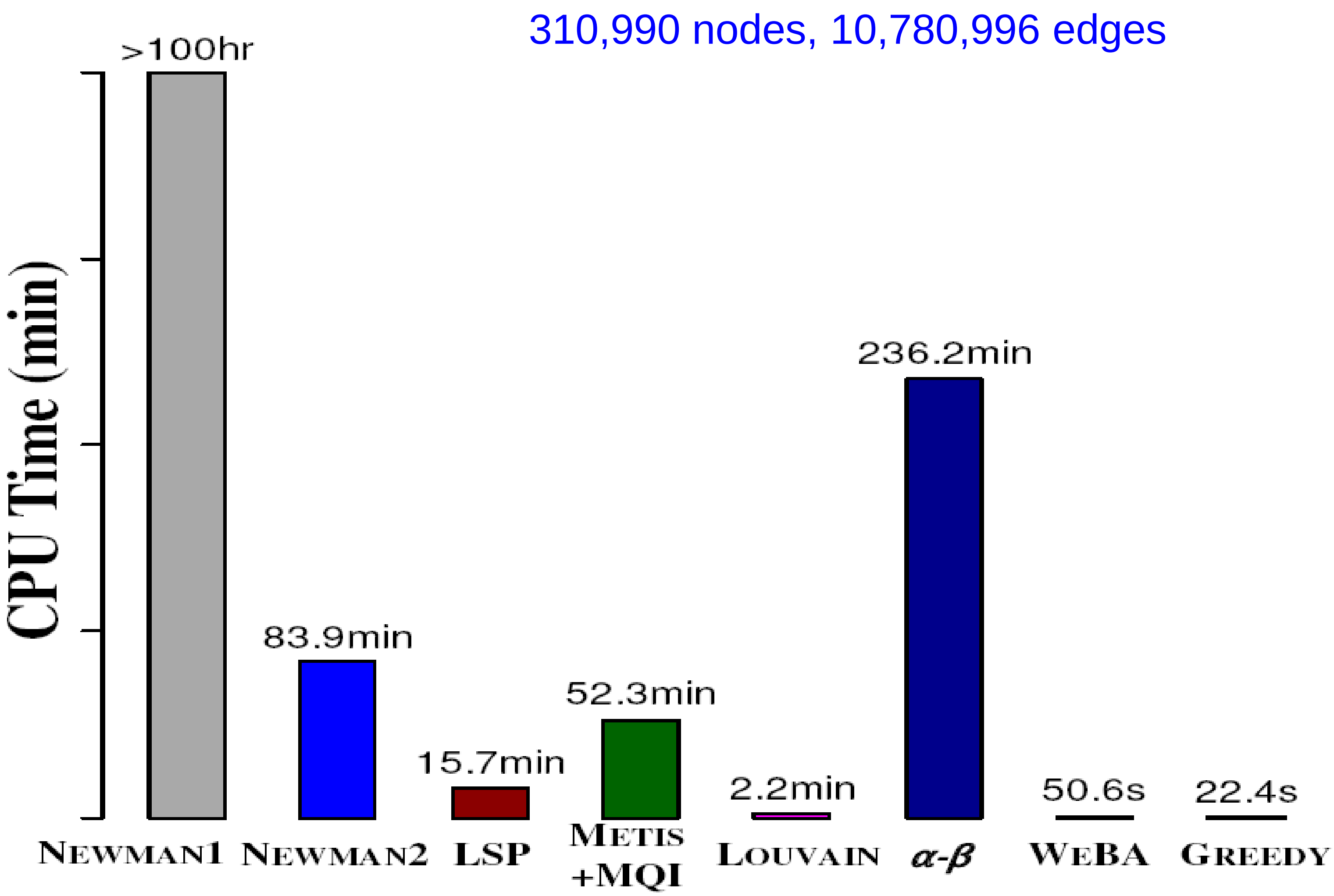


Efficiency — Coauthor

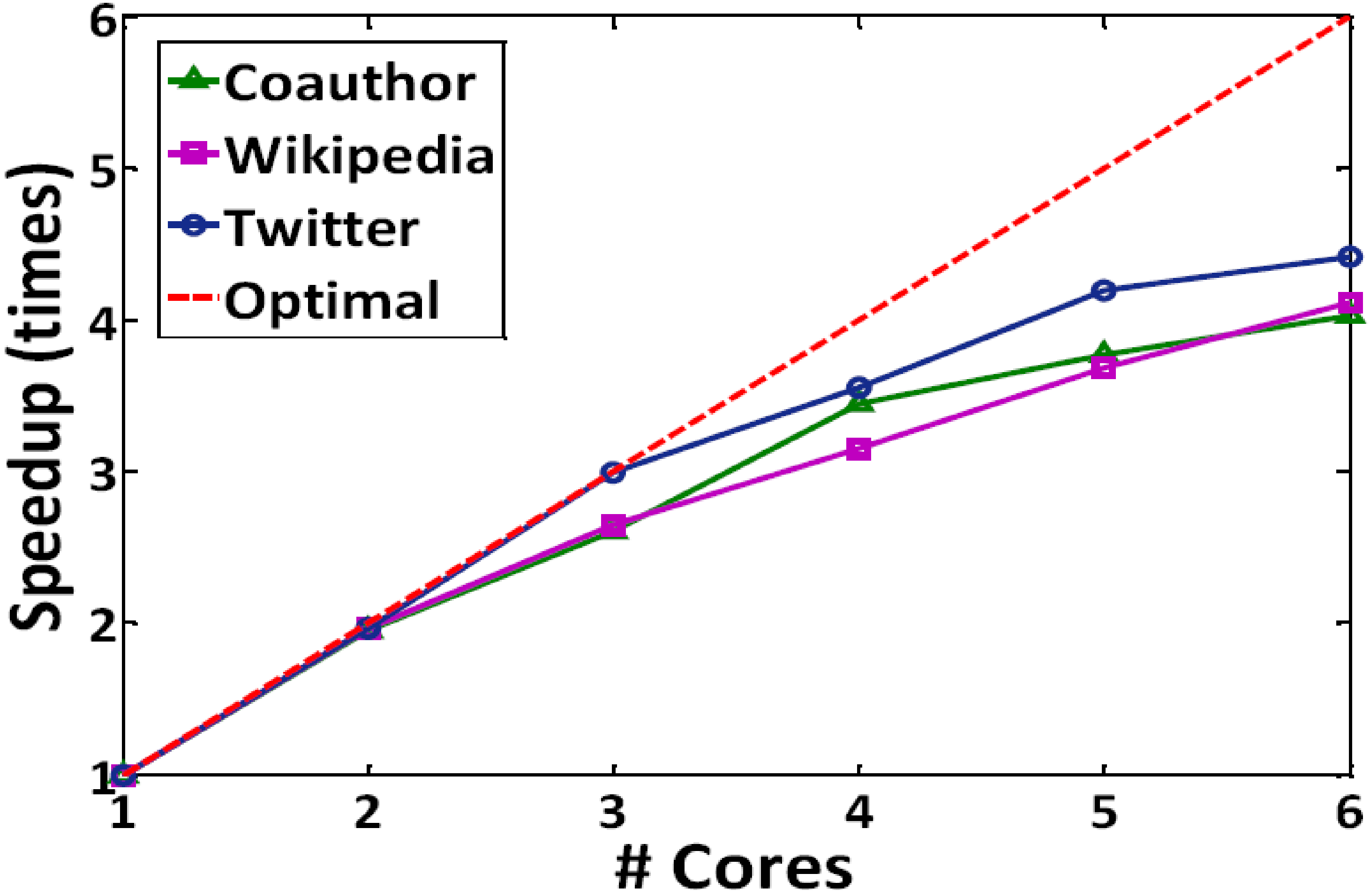
822,415 nodes, 2,928,360 edges



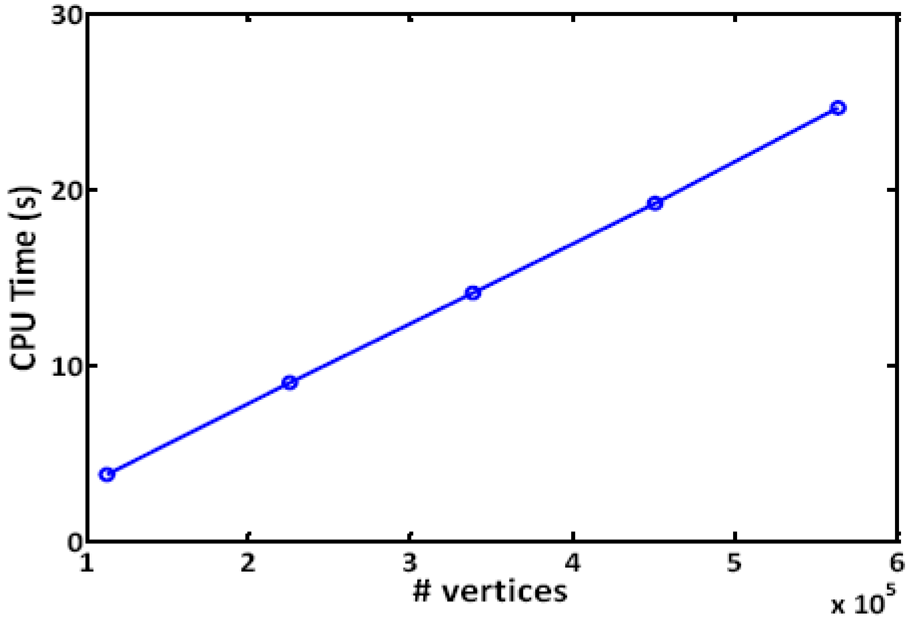
EFFICIENCY — WIKIPEDIA



WEBA — PARALLELIZATION

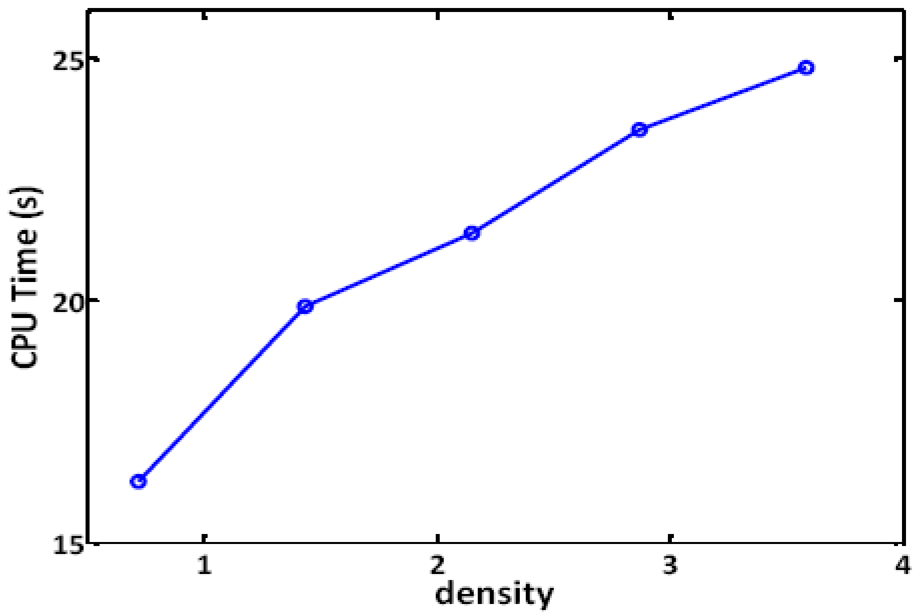


WeBA — Scalability (no parallelization)



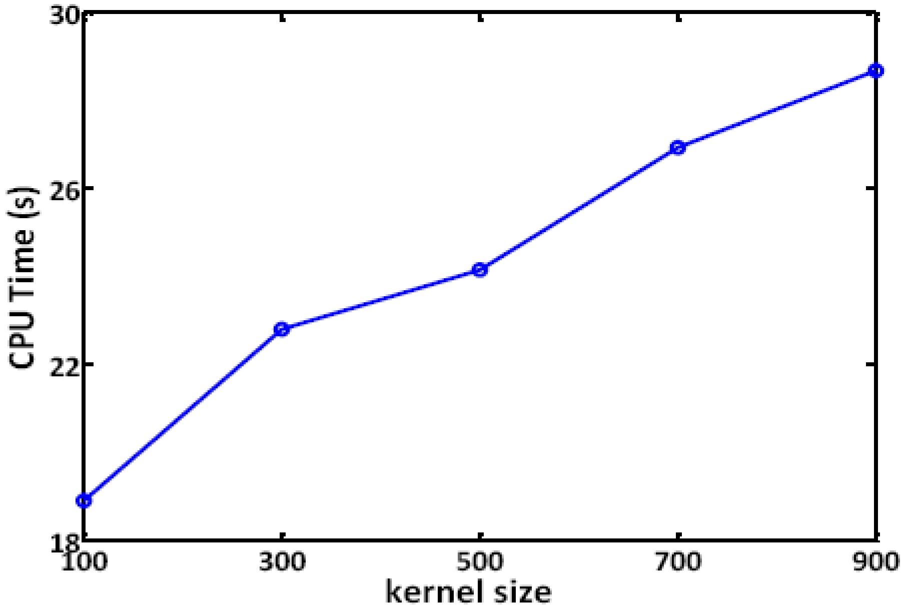
(a) CPU time vs. # vertices

WEBA — SCALABILITY (NO PARALLELIZATION)



(b) CPU time vs. density

WEBA — SCALABILITY (NO PARALLELIZATION)



(c) CPU time vs. kernel size

CONCLUSION

- Structure of community kernels and their auxiliary communities
- Problem definition of detecting community kernels
  - greedy algorithm GREEDY
  - weight-balanced algorithm WEBA (w/ guaranteed error bound)
- WEBA considers both the relative influence of vertices and the link information between auxiliary and kernel members

  ⟹ significantly improves the performance over traditional cut-based and conductance-based algorithms
- WEBA reveals the common profession, interest, or popularity of groups of influential individuals.

THANK YOU!