Sketch-based Influence Maximization and Computation: Scaling up with Guarantees

Edith Cohen   Daniel Delling   *Thomas Pajor*   Renato Werneck

Microsoft Research

4 November 2014

Influence

Influence

- spread of contagion, information diffusion, spread of infection, ...
- Studied in social, biological or physical networks, ...

Applications:

- Viral marketing, product placement [GLM01, RD02],
- sensor placement in water distribution networks for contamination detection [LKG+07],
- ...

Influence

Influence

- spread of contagion, information diffusion, spread of infection, ...
- Studied in social, biological or physical networks, ...

Applications:

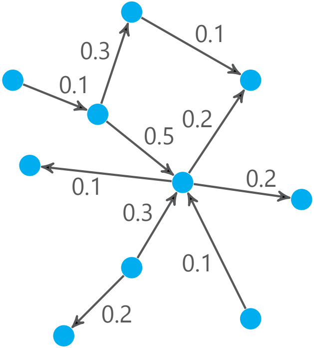
- Viral marketing, product placement [GLM01, RD02],
- sensor placement in water distribution networks for contamination detection [LKG+07],
- ...

Various infection models exist.

Binary Independent Cascade Model [GLM01, KKT03]

Input:

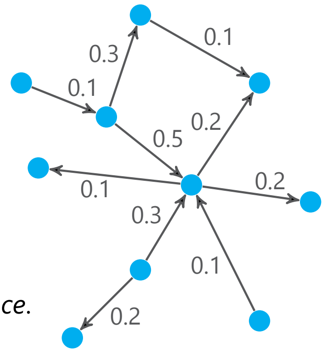
- Directed graph $G = (V, E)$ with
- infection probabilities $p(u, v)$ for every edge $(u, v) \in E$.



Binary Independent Cascade Model [GLM01, KKT03]

Input:

- Directed graph $G = (V, E)$ with
- infection probabilities $p(u, v)$ for every edge $(u, v) \in E$.



Interpretation:

- Edge (u, v) is *live* with probability $p(u, v)$.
- In live case: u is infected $\Rightarrow$ v is infected.
- Set of live edges forms a *propagation instance*.

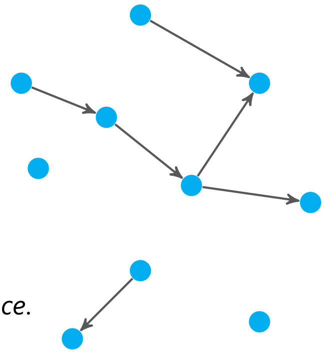
Binary Independent Cascade Model [GLM01, KKT03]

Input:

- Directed graph $G = (V, E)$ with
- infection probabilities $p(u, v)$ for every edge $(u, v) \in E$.

Interpretation:

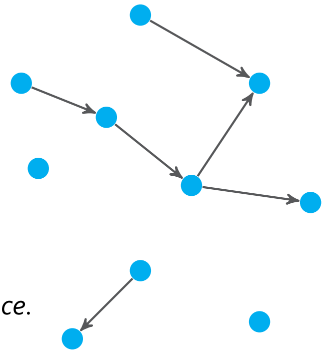
- Edge (u, v) is *live* with probability $p(u, v)$.
- In live case: u is infected $\Rightarrow$ v is infected.
- Set of live edges forms a *propagation instance*.



Binary Independent Cascade Model [GLM01, KKT03]

Input:

- Directed graph $G = (V, E)$ with
- infection probabilities $p(u, v)$ for every edge $(u, v) \in E$.



Interpretation:

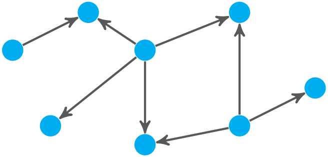
- Edge (u, v) is *live* with probability $p(u, v)$.
- In live case: u is infected $\Rightarrow$ v is infected.
- Set of live edges forms a *propagation instance*.

Definition of Influence

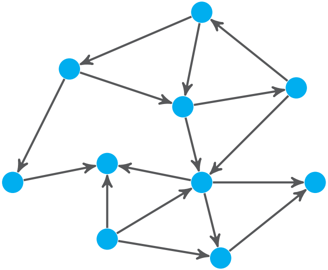
- Given a set S of seed nodes:
- Expected (over prop. instances) number of reachable nodes from S.

Considered Problems

1. **Influence Computation**
   - Given a seed node set *S*:
   - What is the influence of *S* in *G*?

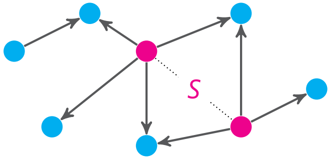


2. **Influence Maximization**
   - Given a number *N*:
   - Compute *sequence S* of seed nodes of length *N* such that
   - influence for every *prefix* of *S* close to maximum for its size.

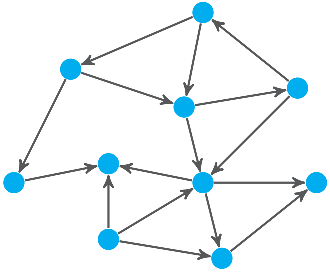


Considered Problems

1. **Influence Computation**
   - Given a seed node set *S*:
   - What is the influence of *S* in *G*?



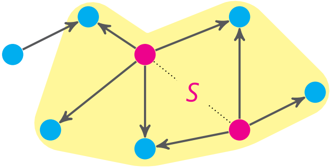
2. **Influence Maximization**
   - Given a number *N*:
   - Compute *sequence S* of seed nodes of length *N* such that
   - influence for every *prefix* of *S* close to maximum for its size.



Considered Problems

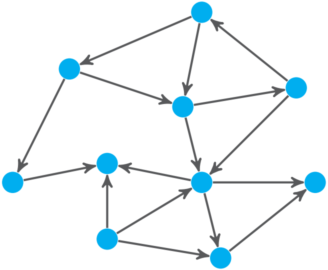
1. Influence Computation

- Given a seed node set *S*:
- What is the influence of *S* in *G*?



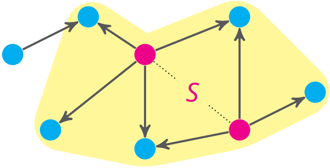
2. Influence Maximization

- Given a number *N*:
- Compute *sequence S* of seed nodes of length *N* such that
- influence for every *prefix* of *S* close to maximum for its size.

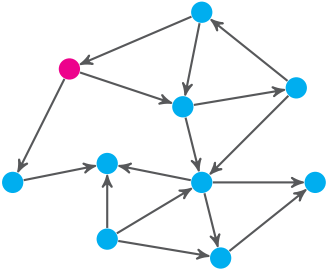


Considered Problems

1. **Influence Computation**
   - Given a seed node set *S*:
   - What is the influence of *S* in *G*?

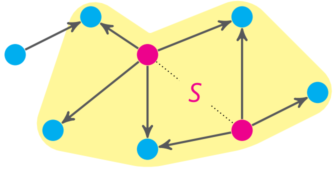


2. **Influence Maximization**
   - Given a number *N*:
   - Compute *sequence* *S* of seed nodes of length *N* such that
   - influence for every *prefix* of *S* close to maximum for its size.

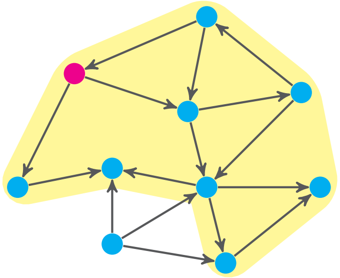


Considered Problems

1. **Influence Computation**
   - Given a seed node set *S*:
   - What is the influence of *S* in *G*?



2. **Influence Maximization**
   - Given a number *N*:
   - Compute *sequence S* of seed nodes of length *N* such that
   - influence for every *prefix* of *S* close to maximum for its size.



Simulation-Based Approach

Problem: Working directly with the probabilistic model is challenging.

Simulation-Based Approach

Problem: Working directly with the probabilistic model is challenging.

Simulation-Based Approach [KKT03]

- Independently draw ℓ propagation instances $G^{(i)}$
  using the given edge probabilities.
- Influence in instance i: # nodes reachable in $G^{(i)}$.
  Can be computed with BFS from S.
- Total Influence: *Average* (over instances) size of reachable sets.

Simulation-Based Approach

Problem: Working directly with the probabilistic model is challenging.

Simulation-Based Approach [KKT03]

- Independently draw ℓ propagation instances $G^{(i)}$
  using the given edge probabilities.
- Influence in instance i: # nodes reachable in $G^{(i)}$.
  Can be computed with BFS from S.
- Total Influence: *Average* (over instances) size of reachable sets.

Properties

- Average influence is unbiased estimate and
- converges to the actual influence.
- Approach also handles *arbitrary* propagation instances;
  e.g., for capturing traces from more complex influence models.

Related Work

Greedy Algorithm [KKT03]

- Uses simulation-based approach.
- In each iteration: Add to S node with maximal *marginal* influence taking into account the current seed nodes.
- Evaluating influence requires graph searches on all instances. Optimizations, such as using lazy evaluation, exist [LKG+07].

$\Rightarrow$ Scales very poorly, even for medium-sized graphs [CWW10].

Related Work

Greedy Algorithm [KKT03]

- Uses simulation-based approach.
- In each iteration: Add to S node with maximal *marginal* influence
  taking into account the current seed nodes.
- Evaluating influence requires graph searches on all instances.
  Optimizations, such as using lazy evaluation, exist [LKG+07].

⇒ Scales very poorly, even for medium-sized graphs [CWW10].

Other Algorithms – usually approximate Greedy, but...

- either have no quality guarantees [CWY09, CWW10, JHC12],
- are not practical [BBCL14],
- or do not compute full *sequence* of max. influence nodes [TXS14].

Related Work

Greedy Algorithm [KKT03]

- Uses simulation-based approach.
- In each iteration: Add to S node with maximal *marginal* influence
  taking into account the current seed nodes.
- Evaluating influence requires graph searches on all instances.
  Optimizations, such as using lazy evaluation, exist [LKG+07].

$\Rightarrow$ Scales very poorly, even for medium-sized graphs [CWW10].

Other Algorithms – usually approximate Greedy, but...

- either have no quality guarantees [CWY09, CWW10, JHC12],
- are not practical [BBCL14],
- or do not compute full *sequence* of max. influence nodes [TXS14].

Existing approaches either slow or without guarantees on quality.

Our Goals

For influence maximization we want an algorithm that...

- computes a *full influence permutation*,
- works with *arbitrary* propagation instances,
- scales well to graphs with *billions of edges*,
- has *guarantees* on the quality.

We want an influence oracle that...

- uses near *linear time preprocessing*
  and near linear storage
- *quickly estimates* for any seed set *S* its influence
- with provably *small relative error*.

Reachability Sketches [Coh97]

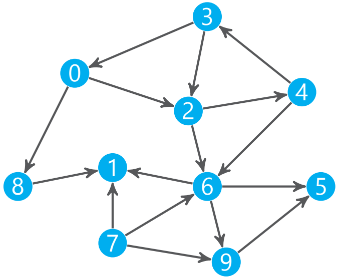
Idea:
Compute small structure per node from which to estimate its influence.

Reachability Sketches [Coh97]

Idea:
Compute small structure per node from which to estimate its influence.

Reachability Sketches:

1. For every node u:
   Assign indep. rank $r(u) \sim U[0,1]$.
2. Sketch $X(u)$:
   k smallest ranks reachable from u.
3. Cardinality est. of u's reachable set given by $(k-1)/\max\{X(u)\}$.

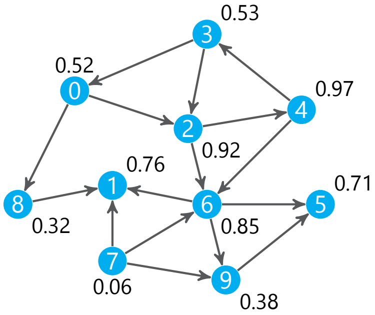


Reachability Sketches [Coh97]

Idea:
Compute small structure per node from which to estimate its influence.

Reachability Sketches:

1. For every node u:
   Assign indep. rank $r(u) \sim U[0, 1]$.
2. Sketch $X(u)$:
   k smallest ranks reachable from u.
3. Cardinality est. of u's reachable set given by $(k-1)/\max\{X(u)\}$.

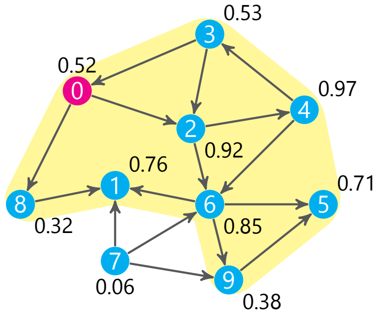


Reachability Sketches [Coh97]

Idea:
Compute small structure per node from which to estimate its influence.

Reachability Sketches:

1. For every node u:
   Assign indep. rank $r(u) \sim U[0,1]$.
2. Sketch $X(u)$:
   k smallest ranks reachable from u.
3. Cardinality est. of u's reachable set given by $(k-1)/\max\{X(u)\}$.



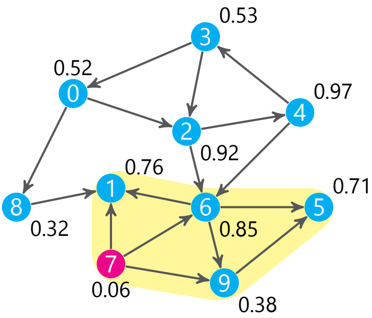
X(0) = {0.32, 0.38, 0.52}

Reachability Sketches [Coh97]

Idea:
Compute small structure per node from which to estimate its influence.

Reachability Sketches:

1. For every node u:
   Assign indep. rank $r(u) \sim U[0, 1]$.
2. Sketch $X(u)$:
   k smallest ranks reachable from u.
3. Cardinality est. of u's reachable set given by $(k-1)/\max\{X(u)\}$.



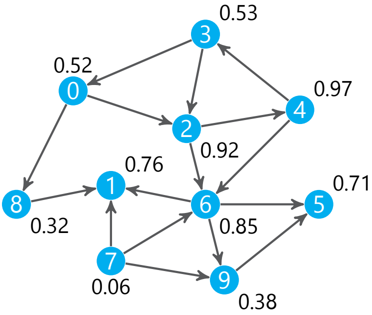
X(0) = {0.32, 0.38, 0.52}
X(7) = {0.06, 0.38, 0.76}

Reachability Sketches [Coh97]

Idea:
Compute small structure per node from which to estimate its influence.

Reachability Sketches:

1. For every node u:
   Assign indep. rank $r(u) \sim U[0,1]$.
2. Sketch $X(u)$:
   k smallest ranks reachable from u.
3. Cardinality est. of u's reachable set given by $(k-1)/\max\{X(u)\}$.



X(0) = {0.32, 0.38, 0.52}
X(7) = {0.06, 0.38, 0.76}

Properties

- Gives unbiased estimate
- with CV of $1/\sqrt{k-2}$,
- which is well concentrated.

Combined Reachability Sketches

Combined Reachability Sketches

- Augments reachability sketches to multiple propagation instances.
- Key difference:
  Assign rank value for every *node/instance* pair $r(u, i) \sim U[0, 1]$.
- Sketch $X(u)$:
  k smallest ranks from reachable sets over all propagation instances.
- Enables estimate on *union* of these reachable sets.

Combined Reachability Sketches

Combined Reachability Sketches

- Augments reachability sketches to multiple propagation instances.
- Key difference:
  Assign rank value for every *node/instance* pair $r(u, i) \sim U[0, 1]$.
- Sketch $X(u)$:
  k smallest ranks from reachable sets over all propagation instances.
- Enables estimate on *union* of these reachable sets.

$\Rightarrow$ Estimated influence of node u using ℓ instances:

$$\widetilde{\mathrm{Inf}}(u) = \frac{1}{\ell} \frac{(k-1)}{\max\{X(u)\}}.$$

Influence Maximization

Sketch-Based Influence Maximization (SKIM)

Paradigm:

1. Approximate greedy algorithm.
2. Use (partial) sketches for influence estimation.

Sketch-Based Influence Maximization (SKIM)

Paradigm:

1. Approximate greedy algorithm.
2. Use (partial) sketches for influence estimation.

Build sketches in reverse fashion:

- For each node/instance pairs (u, i) from smallest to largest rank:
- Run reverse BFS from u in propagation instance $G^{(i)}$.
- Add $r(u, i)$ to sketch $X(v)$ of every scanned node v.

Sketch-Based Influence Maximization (SKIM)

Paradigm:

1. Approximate greedy algorithm.
2. Use (partial) sketches for influence estimation.

Build sketches in reverse fashion:

- For each node/instance pairs (u, i) from smallest to largest rank:
- Run reverse BFS from u in propagation instance $G^{(i)}$.
- Add $r(u, i)$ to sketch $X(v)$ of every scanned node v.

First node v^* for which $|X(v^*)| = k$ has highest marginal influence whp.

Sketch-Based Influence Maximization (SKIM)

Paradigm:

1. Approximate greedy algorithm.
2. Use (partial) sketches for influence estimation.

Build sketches in reverse fashion:

- For each node/instance pairs (u, i) from smallest to largest rank:
- Run reverse BFS from u in propagation instance $G^{(i)}$.
- Add $r(u, i)$ to sketch $X(v)$ of every scanned node v.

First node v^* for which $|X(v^*)| = k$ has highest marginal influence whp.

- Pause sketch building process.
- Return v^* as next node of influence ordering.

Sketch-Based Influence Maximization (SKIM)

Problem: Subsequent nodes must account for *marginal* influence.

Sketch-Based Influence Maximization (SKIM)

Problem: Subsequent nodes must account for *marginal* influence.

Build residual problem:

- Run *forward* BFS from v^* in all propagation instances.
- For every scanned node u in instance i:
- Mark (u, i) as infected, and remove $r(u, i)$ from all sketches.

Sketch-Based Influence Maximization (SKIM)

Problem: Subsequent nodes must account for *marginal* influence.

Build residual problem:

- Run *forward* BFS from v^* in all propagation instances.
- For every scanned node u in instance i:
- Mark (u, i) as infected, and remove $r(u, i)$ from all sketches.

Then: Resume sketch-building process, but skipping infected pairs.

Sketch-Based Influence Maximization (SKIM)

Problem: Subsequent nodes must account for *marginal* influence.

Build residual problem:

- Run *forward* BFS from v^* in all propagation instances.
- For every scanned node u in instance i:
- Mark (u, i) as infected, and remove $r(u, i)$ from all sketches.

Then: Resume sketch-building process, but skipping infected pairs.

Engineering the Algorithm:

- Only maintain *size* of sketches.
  Updated by incrementing/decrementing a *counter* per node.
- Maintain a *reverse index* for each $r(u, i)$
  to enable quickly decrementing relevant counters.

Influence Oracles

Influence Oracles

Two Stage Approach

1. **Preprocessing:**
   Build and store full combined reachability sketches $X(u)$ for all nodes u.
2. **Queries:**
   Estimate influence of S using only sketches $X(u)$ for $u \in S$.

Preprocessing

Reachability Sketches [Coh97]:

- Process nodes u by increasing rank.
- Run reverse BFS from u.
- For each scanned node v:
  If $|X(v)| < k$, add $r(u)$ to $X(v)$, otherwise prune at v.

$\Rightarrow$ Running time is $O(k|V|)$.

Preprocessing

Reachability Sketches [Coh97]:

- Process nodes u by increasing rank.
- Run reverse BFS from u.
- For each scanned node v:
  If $|X(v)| < k$, add $r(u)$ to $X(v)$, otherwise prune at v.

$\Rightarrow$ Running time is $O(k|V|)$.

Combined Reachability Sketches:

- Assign ranks to every vertex/instance pair a priori.
- Compute $X_i(u)$ for each instance i separately.
- Merge k smallest ranks from all $X_i(u)$ into $X(u)$.
- Subsequent computation and merging $\Rightarrow$ $O(k|V|)$ working memory.

Query

Challenge: Given S, must estimate *union* of reachable sets for all $u \in S$.

Query

Challenge: Given S, must estimate *union* of reachable sets for all $u \in S$.

Solution:

- Determine k smallest ranks X from all $X(u)$ for $u \in S$.

$\Rightarrow$ Unbiased cardinality estimate of union: $(k-1)/\max\{X\}$.
Dividing by ℓ gives estimate on influence of S.

- Running time: $O(k|S|)$.

Query

Challenge: Given S, must estimate *union* of reachable sets for all $u \in S$.

Solution:

- Determine k smallest ranks X from all $X(u)$ for $u \in S$.

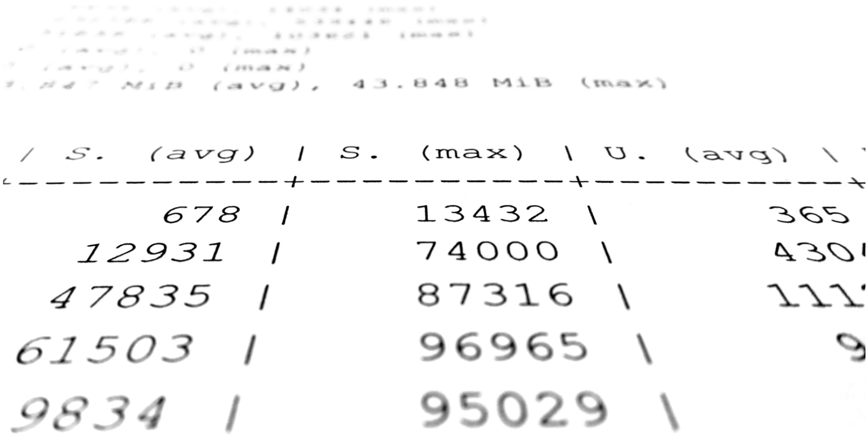
$\Rightarrow$ Unbiased cardinality estimate of union: $(k-1)/\max\{X\}$.
Dividing by ℓ gives estimate on influence of S.

- Running time: $O(k|S|)$.

Improved Estimator:

- Exploit all available rank values (instead of only $\max\{X\}$) [CK09].
- Running time: $O(k|S|\log|S|)$.
- Improves CV by a factor of up to $\sqrt{|S|}$.

Experiments



43.848 MiB (max)

S. (avg)	S. (max)	U. (avg)	
678	13432	365	
12931	74000		
47835	87316		
61503	96965		
9834	95029		

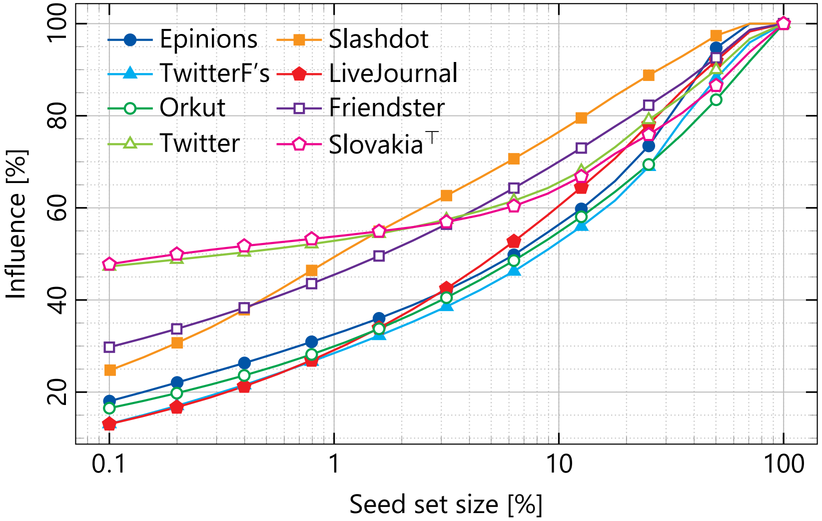
Influence Maximization: SKIM

		Influence [%]		Running time [sec]		
		1000 seeds		1000 seeds		n seeds
Instance	$\|A\|$ $[\cdot 10^3]$	SKIM	IRIE	SKIM	IRIE	SKIM
AstroPh	239.3	45.9	46.5	1.0	4.3	1.9
Epinions	508.8	34.4	34.1	1.6	10.3	6.7
Slashdot	828.2	52.1	52.3	1.9	19.8	7.5
Gowalla	1 900.7	30.9	31.1	3.5	75.2	21.5
TwitterFollowers	14 855.9	17.2	17.5	10.7	388.5	85.1
LiveJournal	68 475.4	6.8	6.7	31.1	4 576.5	933.0
Orkut	234 370.2	12.1	11.5*	102.9	DNF (915)	1 197.2
Friendster	1 806 067.1	15.4	8.8*	1 308.5	DNF (43)	19 254.2
Twitter	1 468 364.9	38.0	25.3*	1 912.8	DNF (92)	11 558.8
Slovakia	1 930 292.9	25.9	16.7*	621.4	DNF (230)	11 679.3

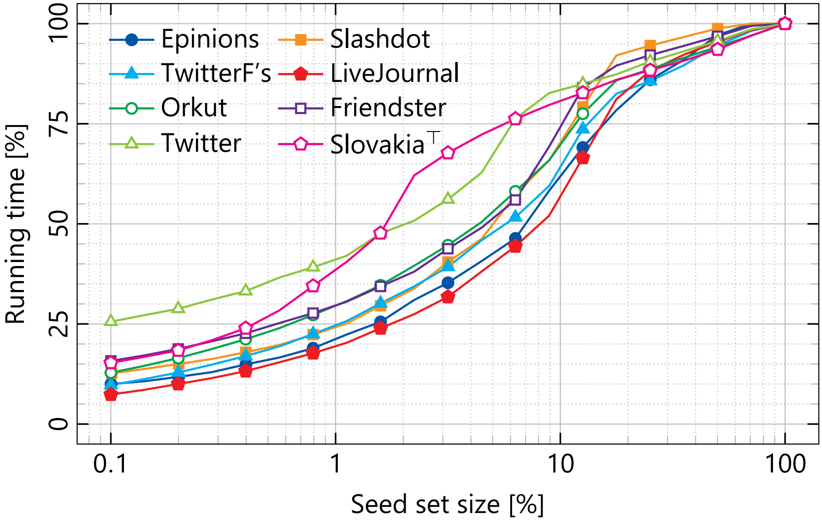
Parameters: $k = 64$, $\ell = 64$. Machine: 1 core of Xeon E5-2690 @ 2.9 GHz; 384 GiB RAM.

IRIE $\equiv$ state-of-the-art heuristic [JHC12].
DNF $\equiv$ does not finish within 2 hours.

Full Influence Permutations: Influence



Full Influence Permutations: Running Time



Influence Oracle

	Preproc.		Queries					
			1 Seed		50 Seeds		1000 seeds	
Instance	Time [sec]	Space [MiB]	Time [µs]	Err. [%]	Time [µs]	Err. [%]	Time [µs]	Err. [%]
AstroPh	4	7.2	1.6	8.5	166.7	2.1	4 658.3	0.5
Epinions	10	37.1	1.3	5.2	155.0	3.4	5 011.1	1.1
Slashdot	20	37.8	1.5	6.0	155.2	3.9	4 982.3	1.0
Gowalla	46	96.0	1.5	7.3	179.8	3.2	5 275.6	1.1
TwitterFollowers	229	223.0	2.1	7.0	190.2	3.3	5 061.8	0.8
LiveJournal	2 064	2 367.0	2.0	7.1	189.6	3.0	5 168.3	0.9

Parameters: $k = 64$, $\ell = 64$. Machine: 1 core of Xeon E5-2690 @ 2.9 GHz; 384 GiB RAM.

Conclusion

Influence Maximization: SKIM

- Simple algorithm to compute *full influence permutation* of nodes.
- Exploits theory of *combined reachability sketches*.
- Every prefix *approximates maximum influence* for its size.
- *Fast and Practical:* Scales to graphs with billions of edges.
- Can be extended to *adaptively* set *k* for given error bound.
  See paper for details.

Conclusion

Influence Maximization: SKIM

- Simple algorithm to compute *full influence permutation* of nodes.
- Exploits theory of *combined reachability sketches*.
- Every prefix *approximates maximum influence* for its size.
- *Fast and Practical:* Scales to graphs with billions of edges.
- Can be extended to *adaptively* set *k* for given error bound.
  See paper for details.

Influence Oracle:

- Computes *combined reachability sketches* for all nodes.
- *Influence estimation* for sets of seed nodes form sketches only.
- *Fast and practical:* Preprocessing in minutes–hours; queries in µs–ms.

Thank you!

Paper at:

`http://arxiv.org/abs/1408.6282`

Bibliography I

C. Borg, M. Brautbar, J. Chayes, and B. Lucier.
Maximizing social influence in nearly optimal time.
In *SODA*, 2014.

E. Cohen and H. Kaplan.
Leveraging discarded samples for tighter estimation of multiple-set aggregates.
In *ACM SIGMETRICS*, 2009.

E. Cohen.
Size-estimation framework with applications to transitive closure and reachability.
Journal of Computer and System Sciences, 55:441–453, 1997.

W. Chen, C. Wang, and Y. Wang.
Scalable influence maximization for prevalent viral marketing in large-scale social networks.
In *KDD*. ACM, 2010.

W. Chen, Y. Wang, and S. Yang.
Efficient influence maximization in social networks.
In *KDD*. ACM, 2009.

J. Goldenberg, B. Libai, and E. Muller.
Talk of the network: A complex systems look at the underlying process of word-of-mouth.
Marketing Letters, 12(3), 2001.

K. Jung, W. Heo, and W. Chen.
Irie: Scalable and robust influence maximization in social networks.
In *ICDM*. ACM, 2012.

Bibliography II

D. Kempe, J. M. Kleinberg, and É. Tardos.
Maximizing the spread of influence through a social network.
In *KDD*. ACM, 2003.

J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and Glance N.
Cost-effective outbreak detection in networks.
In *KDD*. ACM, 2007.

M. Richardson and P. Domingos.
Mining knowledge-sharing sites for viral marketing.
In *KDD*. ACM, 2002.

Y. Tang, X. Xiao, and Y. Shi.
Influence maximization: Near-optimal time complexity meets practical efficiency.
In *SIGMOD*, 2014.