2017/5/17 SIGMOD'17@Chicago

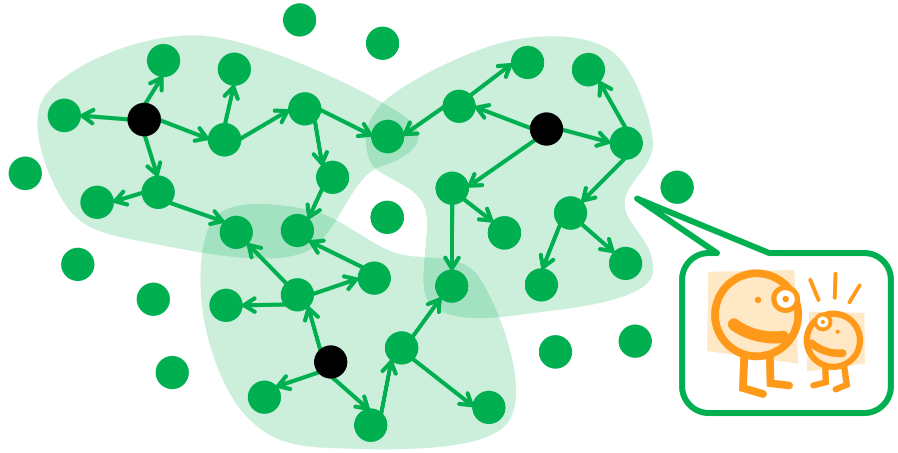
Coarsening Massive Influence Networks

for Scalable Diffusion Analysis

Naoto Ohsaka (UTokyo)
Tomohiro Sonobe (NII)
Sumio Fujita (Yahoo Japan Corp.)
Ken-ichi Kawarabayashi (NII)
Kawarabayashi Large Graph Project

Social network diffusion

A prime medium of information dissemination



Q. How to find the most influential group?

Marketing strategies [Domingos-Richardson. *KDD'01*]

||

Influence maximization

[Kempe-Kleinberg-Tardos. *KDD'03*]

Algorithmic problem on *influence graphs*

Diffusion analysis at scale

Effort on influence maximization methods

[*KDD'03*] [*KDD'07*] [*AAAI'07*] [*KDD'09*] [*KDD'10*] [*WWW'11*] [*ICDM'12*] [*CIKM'13*] [*SODA'14*] [*AAAI'14*] [*SIGMOD'14*] [*CIKM'14*] [*SIGIR'14*] [*SIGMOD'15*] [*SIGMOD'16*] ...

But ...

300M users & 60B links

1.4B users & 400B links

No single state-of-the-art

[Arora-Galhotra-Ranu. *SIGMOD'17*] (*next talk*)

Our goal :

Scalable diffusion analysis via *graph reduction*

Studies on graph reduction

Reduce the size while preserving a *certain* property

- Reachability [Zhou-Zhou-Yu-Wei-Chen-Tang. *SIGMOD'17*]
- Clustering results [Satuluri-Parthasarathy-Ruan. *SIGMOD'11*]
- Personalized PageRank [Vattani-Chakrabarti-Gurevich. *ICML'11*]
- Edge cuts [Benczur-Karger. *STOC'96*]
- Spectral properties [Spielman-Teng. *STOC'04*]

☹ Do not preserve diffusion properties

Reduction methods for *influence graphs*

- SPINE [Mathioudakis-Bonchi-Castillo-Gionis-Ukkonen. *KDD'11*]
- COARSENET [Purohit-Prakash-Kang-Zhang-Subrahmanian. *KDD'14*]

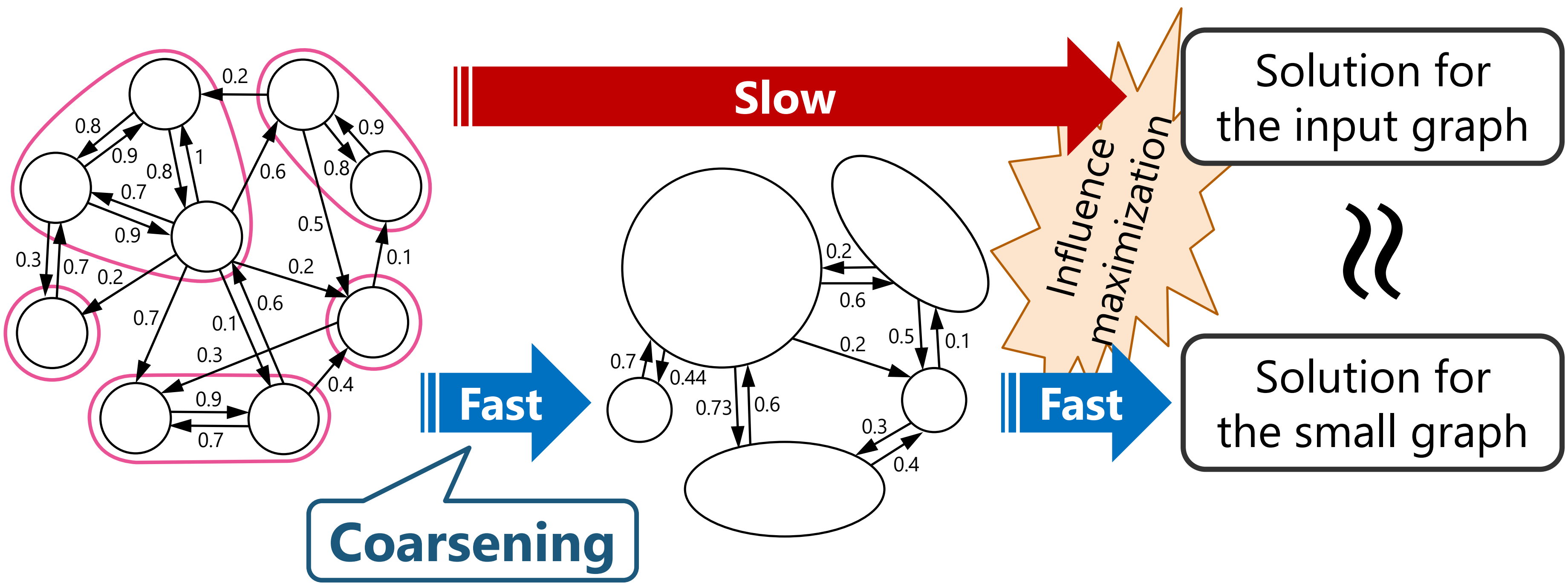
☹ Low scalability & no quality guarantee

Our contribution

We propose

reduction strategy, scalable algorithm, analysis frameworks

Accuracy guarantee 1 hour for billion edges 2 – 30× faster

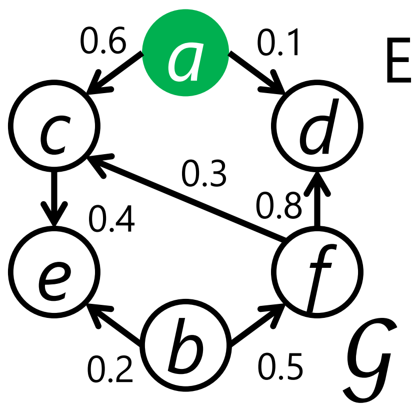


Independent cascade diffusion model

[Goldenberg-Libai-Muller. *Market. Lett.*'01]

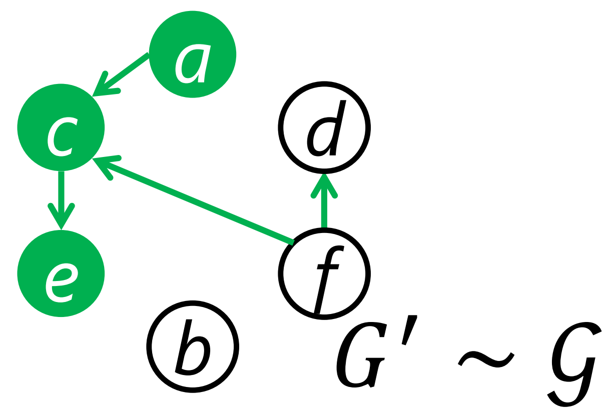
▶ Influence graph $\mathcal{G} = (V, E, p)$ & Seed set $S \subseteq V$

Diffusion process on $\mathcal{G}$ **=** Reachability on the random graph $G' \sim \mathcal{G}$



Edge e lives w.p. p_e

$2^{|E|}$ outcomes



Influence spread

$$\mathrm{Inf}_{\mathcal{G}}(S) = \mathbf{E}_{G' \sim \mathcal{G}}\left[\text{\# vertices reachable from } S \text{ in } G'\right]$$

[Kempe-Kleinberg-Tardos. *KDD*'03]

Two influence analysis problems

Influence estimation

Input seed set S

Output $\mathrm{Inf}_{\mathcal{G}}(S)$

− #P-hard [Chen-Wang-Wang. *KDD'10*]

+ Monte-Carlo is good approx.

Repeat random graph generation

Influence maximization

[Kempe-Kleinberg-Tardos. *KDD'03*]

Input integer k

Output $\underset{S:|S|=k}{\mathrm{argmax}}\ \mathrm{Inf}_{\mathcal{G}}(S)$

− NP-hard [Kempe+'03]

+ Greedy strategy is $(1 - \mathrm{e}^{-1}) \approx$ 63%-approx.

[Nemhauser-Wolsey-Fisher. *Math. Program.'78*]

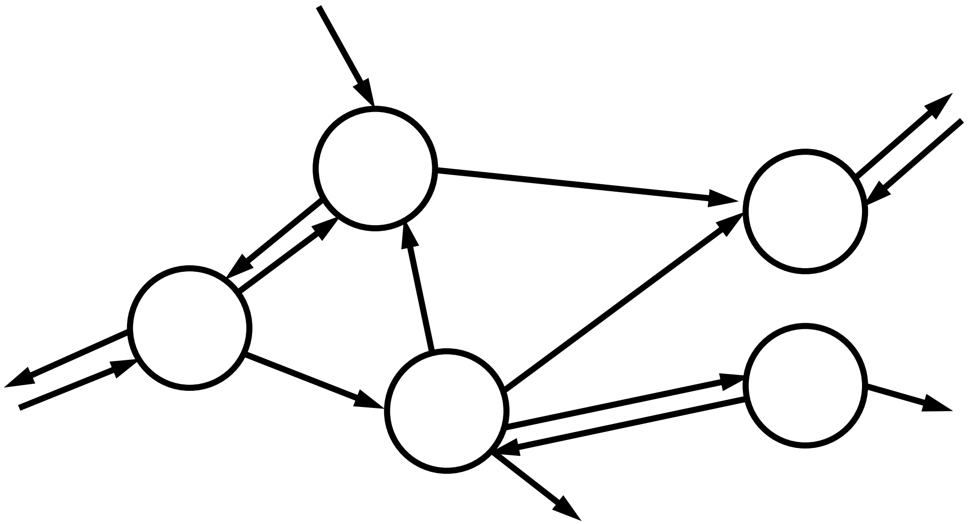
$\mathrm{Inf}_{\mathcal{G}}(\cdot)$ is submodular [Kempe+'03]

Computation cost ≈ Edge traversal cost

Design concept (1)

Our central idea = **Coarsening**

Make no distinction among vertices in a certain set



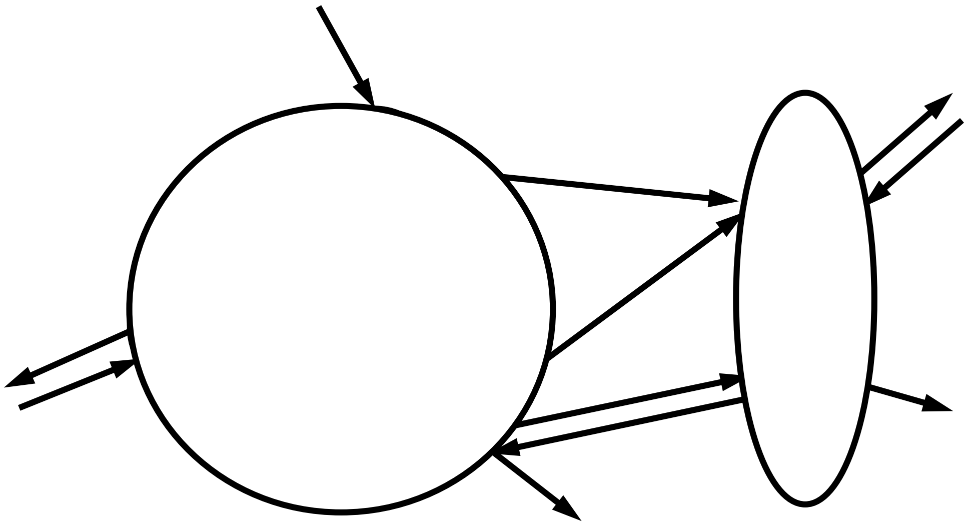
+ Potential to great reduction of # edges

Our strategy

Design concept (2)

Our central idea = **Coarsening**

Make no distinction among vertices in a certain set

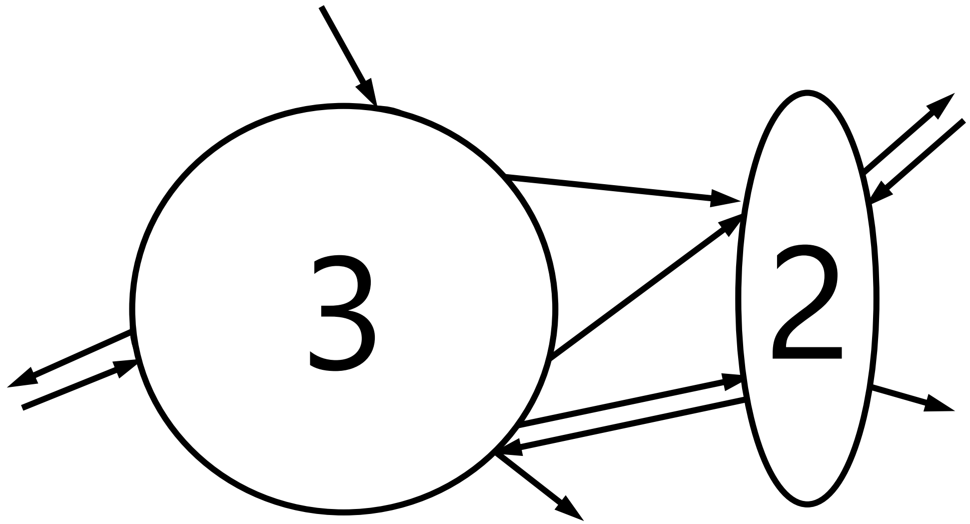


+ Potential to great reduction of # edges

Design concept (3)

Our central idea = **Coarsening**

Make no distinction among vertices in a certain set



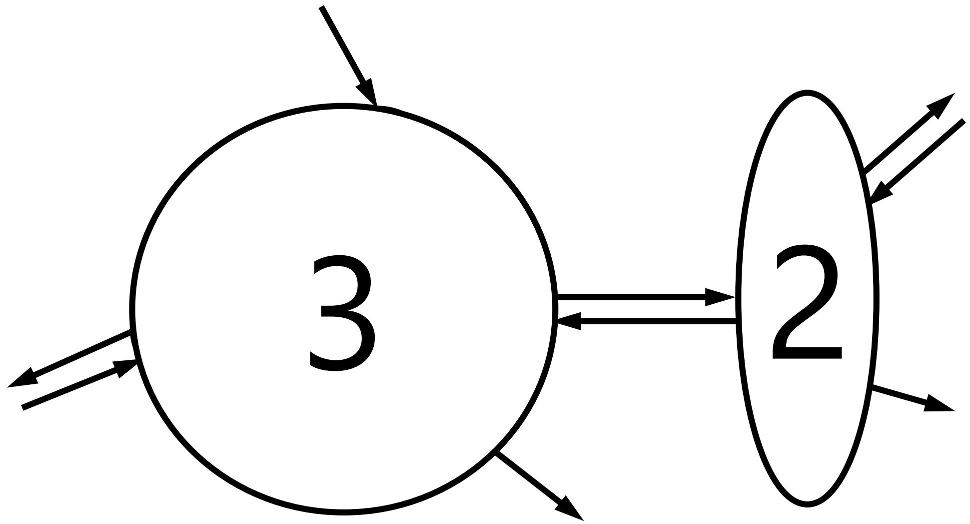
+ Potential to great reduction of # edges

Our strategy

Design concept (4)

Our central idea = **Coarsening**

Make no distinction among vertices in a certain set

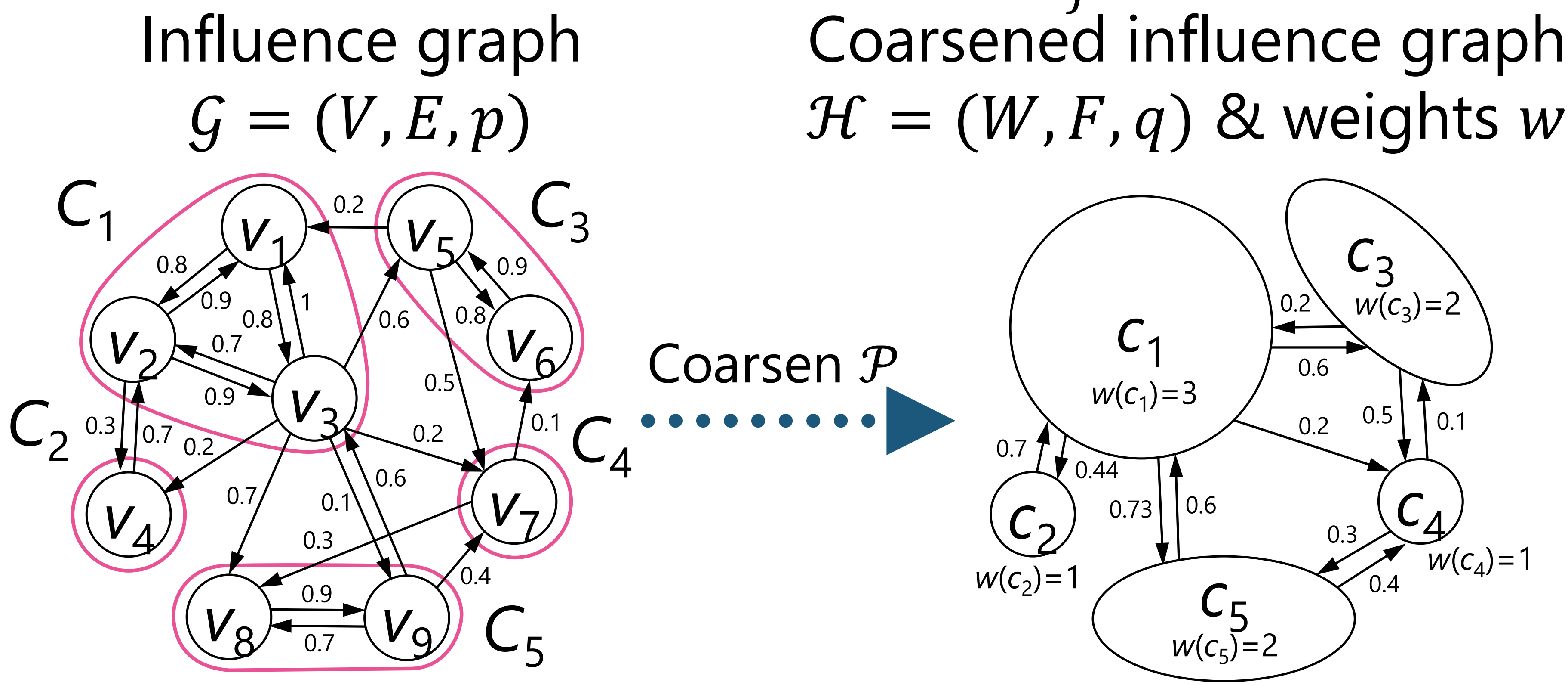


+ Potential to great reduction of # edges

Our strategy

Coarsened influence graphs

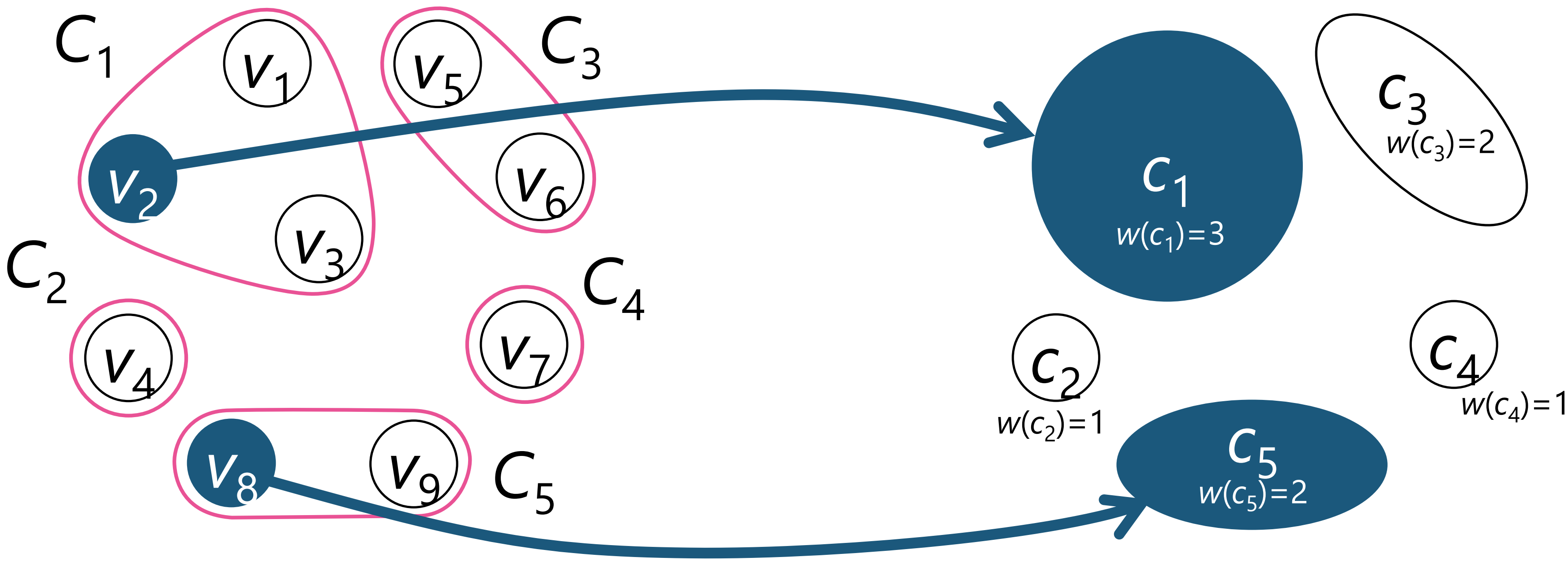
We specify a vertex partition $\mathcal{P} = \{C_j\}_j$



Coarsened influence graphs

We specify a vertex partition $\mathcal{P} = \{C_j\}_j$

Influence graph $\mathcal{G} = (V, E, p)$ | Coarsened influence graph $\mathcal{H} = (W, F, q)$ & weights w



Vertex in C_j $\mapsto$ Weighted vertex c_j

$|C_j| = w(c_j)$

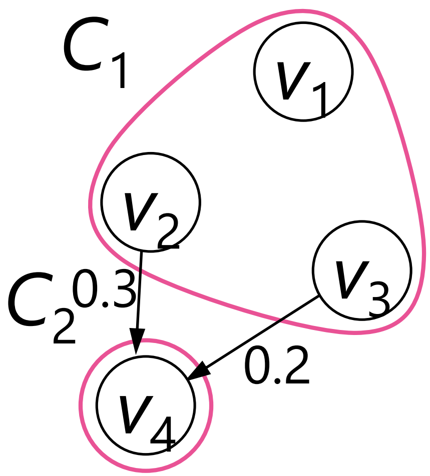
Our strategy

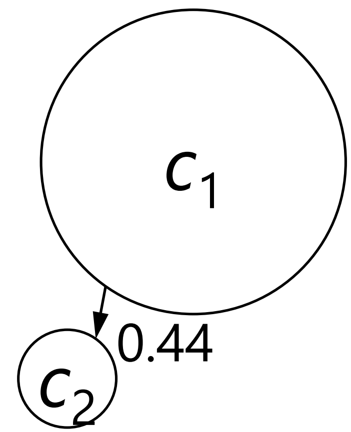
Coarsened influence graphs

We specify a vertex partition $\mathcal{P} = \{C_j\}_j$

Influence graph $\mathcal{G} = (V, E, p)$

Coarsened influence graph $\mathcal{H} = (W, F, q)$ & weights w





$$\mathbf{Pr}[v_2v_4 \text{ lives OR } v_3v_4 \text{ lives}] = \mathbf{Pr}[c_1c_2 \text{ lives}]$$

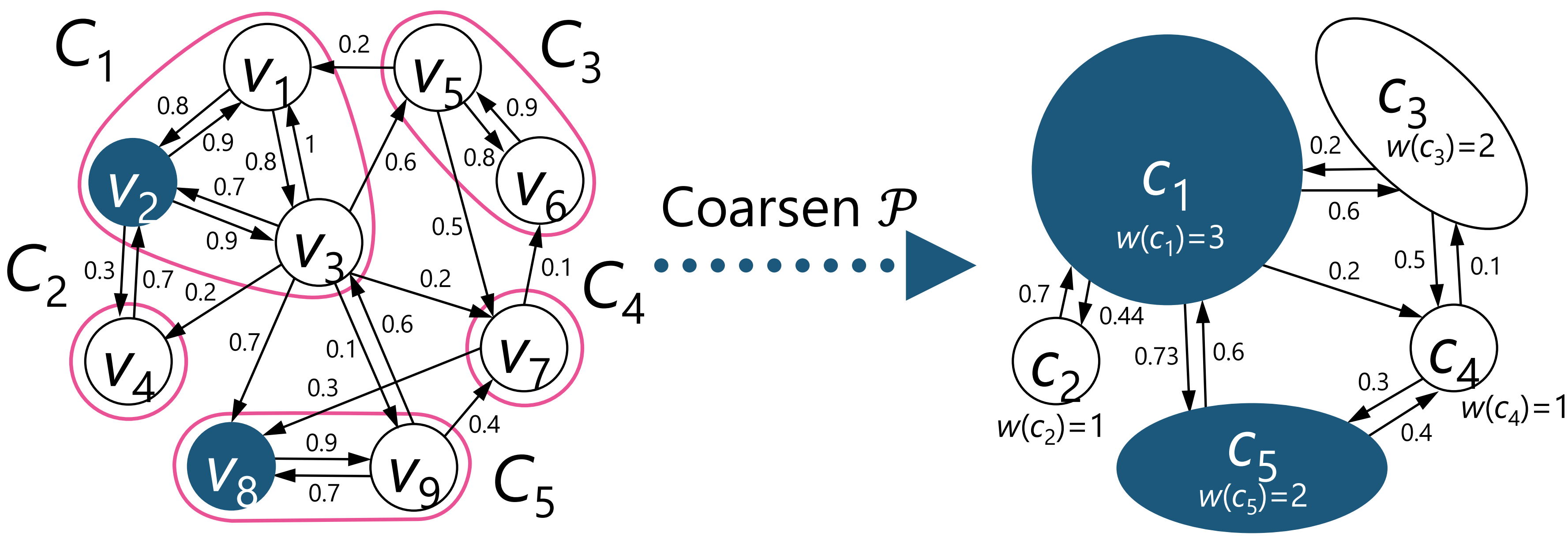
$$1 - (1 - p_{v_2v_4})(1 - p_{v_3v_4}) = q_{c_1c_2}$$

$$1 - (1 - 0.3)(1 - 0.2) = 0.44$$

Coarsened influence graphs

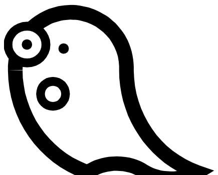
We specify a vertex partition $\mathcal{P} = \{C_j\}_j$

Influence graph $\mathcal{G} = (V, E, p)$ | Coarsened influence graph $\mathcal{H} = (W, F, q)$ & weights w



Wish : $\mathrm{Inf}_{\mathcal{G}}(\{v_2, v_8\}) \approx \mathrm{Inf}_{\mathcal{H}}(\{c_1, c_5\})$

So, what is a *good* partition?



Our strategy

Gap of influence between $\mathcal{G}$ and $\mathcal{H}$

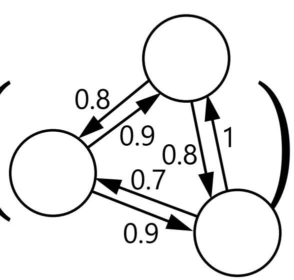
Theorem 4.6

The lower the better

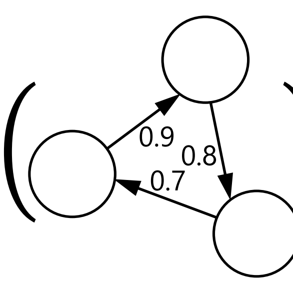
$$\mathrm{Inf}_{\mathcal{G}}(\cdot) \leq \mathrm{Inf}_{\mathcal{H}}(\cdot) \leq \frac{1}{\prod_{C_j \in \mathcal{P}} \mathrm{Rel}\left(\mathcal{G}[C_j]\right)} \cdot \mathrm{Inf}_{\mathcal{G}}(\cdot)$$

$\mathrm{Rel}(\mathcal{G}) \coloneqq \mathbf{Pr}_{G' \sim \mathcal{G}}[G' \text{ is strongly connected}]$ (called *reliability*)

$\mathcal{G}[C_j] \coloneqq$ subgraph of $\mathcal{G}$ induced by C_j

$\mathrm{Rel}\Bigg($ 

$\Bigg) = 0.88848$

$\mathrm{Rel}\Bigg($ 

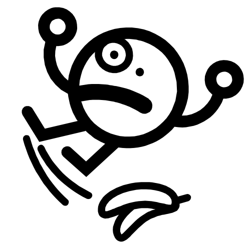
$\Bigg) = 0.504$

Our strategy

Our answer for a good partition

We want a partition $\mathcal{P}$ with high $\prod_{C_j \in \mathcal{P}} \mathrm{Rel}(\mathcal{G}[C_j])$

- Exact computation of $\mathrm{Rel}(\cdot)$ is #P-hard
  [Valiant. *SIAM J. Comput.*'79] [Ball. *Networks*'80]
- Approximate computation needs a large # samples



Our insight

We need high $\mathrm{Rel}(\cdot)$ vertex sets only, so how about using just a *small* # samples?

We introduce **r-robust SCCs**

strongly connected components

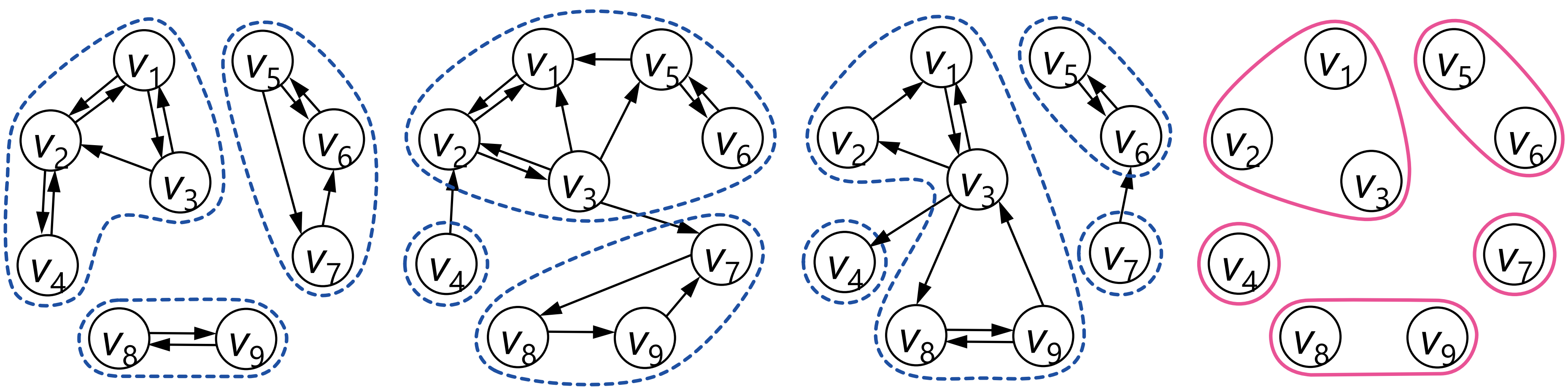
Definition of r-robust SCCs

C is an **r-robust SCC** w.r.t. r subgraphs $G_1, \ldots, G_r$ if

① C is strongly connected in every G_i

② C is maximal

\+ No need to estimate $\mathrm{Rel}(\mathcal{G}[C])$

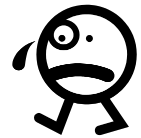


Subgraph G_1 Subgraph G_2 Subgraph G_3 3-robust SCCs

Sampled from $\mathcal{G}$ by keeping edge e w.p. p_e

Our strategy

Limitation & advantages of r-robust SCCs

No bound on $\prod_{C_j \in \mathcal{P}} \mathrm{Rel}(\mathcal{G}[C_j])$ 

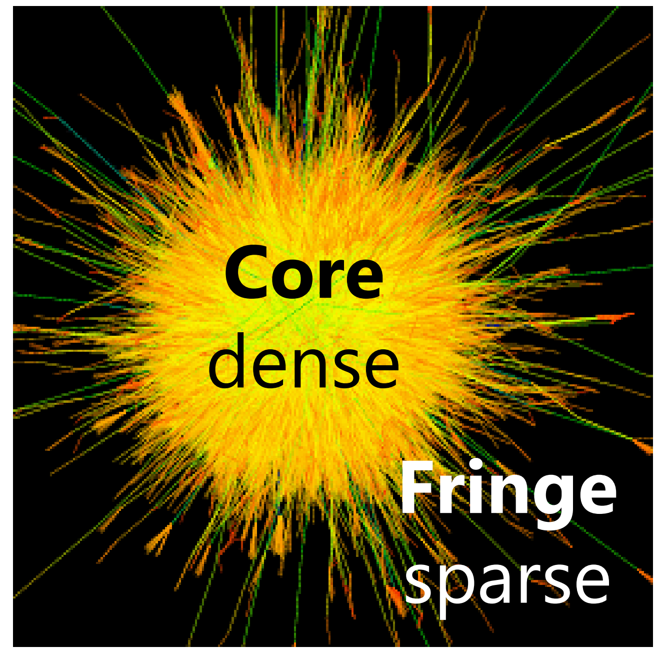
$\mathcal{P}$:= collection of r-robust SCCs

Justification from a theoretical point of view

Theorem 4.12
They *include* high $\mathrm{Rel}(\cdot)$ vertex sets
⇝ Expected to preserve $\mathrm{Inf}(\cdot)$

Theorem 4.13
They are *dense*
⇝ Great reduction of # edges

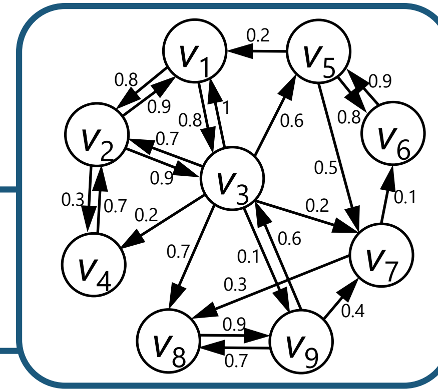
Core-fringe structure
[Leskovec-Lang-Dasgupta-Mahoney. *WWW'08*]
[Maehara-Akiba-Iwata-Kawarabayashi. *PVLDB'14*]



http://www.cise.ufl.edu/research/sparse/matrices/SNAP/soc-Epinions1.html

Our algorithm

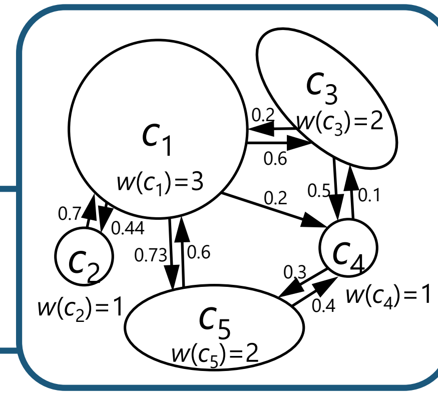
Input : $\mathcal{G} = (V, E, p)$ & r



Stage 1 : Extract r-robust SCCs

Stage 2 : Coarsen each r-robust SCC

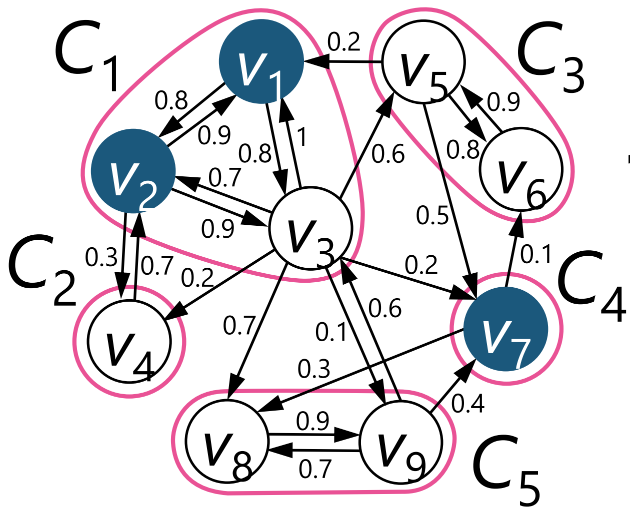
Output : $\mathcal{H} = (W, F, q)$ & w



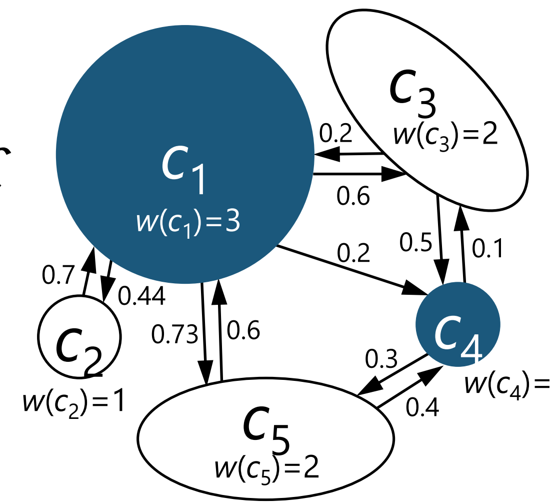
Speed-oriented	$O(r(\|V\| + \|E\|))$ time	$O(\|V\| + \|E\|)$ space
Scalability-oriented Disk-based SCC algorithms Space reduction technique	$O(r(\|V\| + \|E\|))$ time in practice	$O(\|V\| + \|F'\|)$ space $\|F'\| \ll \|F\|$ in practice

Influence estimation framework

Task : $\mathrm{Inf}_{\mathcal{G}}(S)$



1. Map S onto $\mathcal{H}$

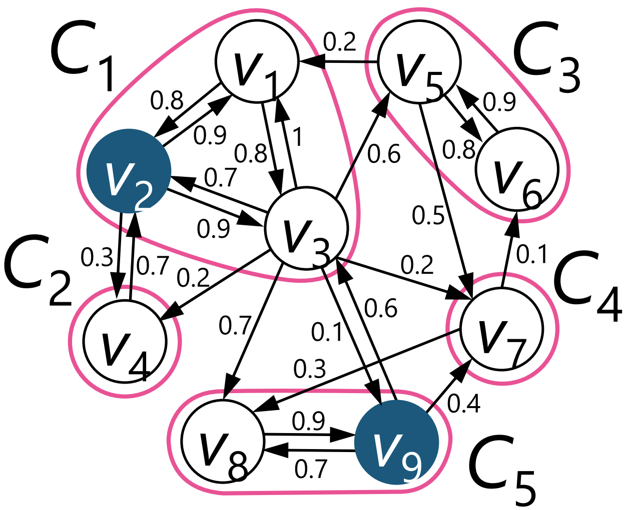


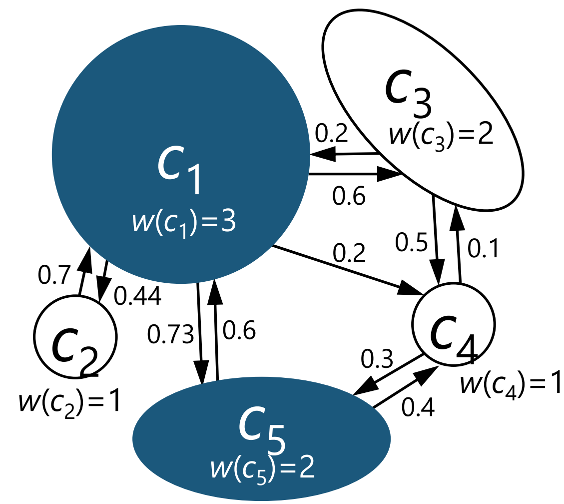
2. Estimate $\mathrm{Inf}_{\mathcal{H}}(T)$ using *existing methods*

Est. of $\mathrm{Inf}_{\mathcal{G}}(\{v_1, v_2, v_7\})$ $\approx$ Est. of $\mathrm{Inf}_{\mathcal{H}}(\{c_1, c_4\})$

Influence maximization framework

Task : $\underset{S:|S|=k}{\operatorname{argmax}}\ \mathrm{Inf}_{\mathcal{G}}(S)$





1. Extract T of size k from $\mathcal{H}$ using *existing methods*

$S = \{v_2, v_9\}$

$T = \{c_1, c_5\}$

2. Map T onto $\mathcal{G}$

Setup

Used social, communication, and web graphs from

Laboratory for Web Algorithmics, Stanford Network Analysis Project, Yahoo Japan Corp.

Probability setting

- exponential $\sim \exp(0.1)$ — Motivated by [Barbieri+'12] [Dickens+'12]
- trivalency $\sim \{0.1, 0.01, 0.001\}$ — [Chen+'10]
- weighted $= (\text{indegree})^{-1}$ — [Kempe+'03]
- uniform $= 0.1$ — [Kempe+'03]

(trivalency, weighted, uniform: see our paper)

Algorithm settings

- $r = 16$ (default)
- Use a disk-based SCC algorithm of [Laura-Santaroni. *TAPAS*'11]

Environment

- Intel Xeon E5-2690 2.90GHz CPU + 256GB memory & g++v4.6.3

Experiments

Run time & memory usage

dataset	$\|V\|$	$\|E\|$	speed-oriented		scalability-oriented	
			run time	memory usage	run time	memory usage
soc-Slashdot0922	0.1M	0.9M	< 1s	< 1GB	6s	< 1GB
wiki-Talk	2M	5M	42s	< 1GB	57s	< 1GB
soc-Pokec	2M	31M	35s	1GB	224s	< 1GB
soc-LiveJournal1	5M	68M	95s	3GB	508s	1GB
twitter-2010	42M	1,468M	1,763s	50GB	11,522s	6GB
com-Friendster	66M	3,612M	3,964s	101GB	26,424s	8GB
uk-2007-05	105M	3,717M	3,106s	137GB	29,540s	11GB
ameblo	273M	6,910M	—	OOM	35,761s	28GB

10× slower

10× smaller

Scale to large graphs

Time & space $\propto |E|$

Graph size reduction

$\mathcal{G} = (V, E, p)$ input/original graph

$\mathcal{H} = (W, F, q)$ output/coarsened graph

dataset	$\lvert V \rvert$	$\lvert E \rvert$	$\lvert W \rvert / \lvert V \rvert$ ≫	$\lvert F \rvert / \lvert E \rvert$
soc-Slashdot0922	0.1M	0.9M	95.2%	36.0%
wiki-Talk	2M	5M	99.8%	61.4%
soc-Pokec	2M	31M	89.0%	43.4%
soc-LiveJournal1	5M	68M	92.8%	42.2%
twitter-2010	42M	1,468M	93.2%	23.5%
com-Friendster	66M	3,612M	71.2%	4.7%
uk-2007-05	105M	3,717M	97.3%	41.8%
ameblo	273M	6,910M	99.4%	79.3%

Achieved great reduction of # edges

There is a giant & dense r-robust SCC

Influence estimation framework

Apply our framework to *Monte-Carlo simulations*

Run the diffusion process from a random vertex 10,000 times

dataset	run time			edge reduction
	Monte-Carlo	Our framework w/ *Monte-Carlo*	time reduction ≈	
soc-Slashdot0922	32s	8s	25.4%	36.0%
wiki-Talk	11s	7s	63.7%	61.4%
soc-Pokec	2,442.3s	897.1s	36.7%	43.4%
soc-LiveJournal1	5,349s	1,783s	33.3%	42.2%
twitter-2010	106,428s	24,212s	22.8%	23.5%
com-Friendster	540,483s	18,968s	3.5%	4.7%
uk-2007-05	5,719s	1,900s	33.2%	41.8%

Our framework's estimations are accurate (see our paper)

mean average relative error ≤ 0.1 & rank correlation coefficient ≥ 0.88

Influence maximization framework

Apply our framework to *D-SSA* [Nguyen-Thai-Dinh. *SIGMOD'16*]

Extract a seed set of size 100

dataset	$\lvert E \rvert$	run time		
		D-SSA	Our framework w/ *D-SSA*	time reduction
soc-Slashdot0922	0.9M	141 s	79 s	56.1%
wiki-Talk	5M	522 s	155 s	29.7%
soc-Pokec	31M	18,350s	6,216s	33.9%
soc-LiveJournal1	68M	OOM	OOM	—
twitter-2010	1,468M	OOM	OOM	—
com-Friendster	3,612M	OOM	OOM	—
uk-2007-05	3,717M	OOM	OOM	—

Our framework's solutions are comparable to *D-SSA* (see our paper)

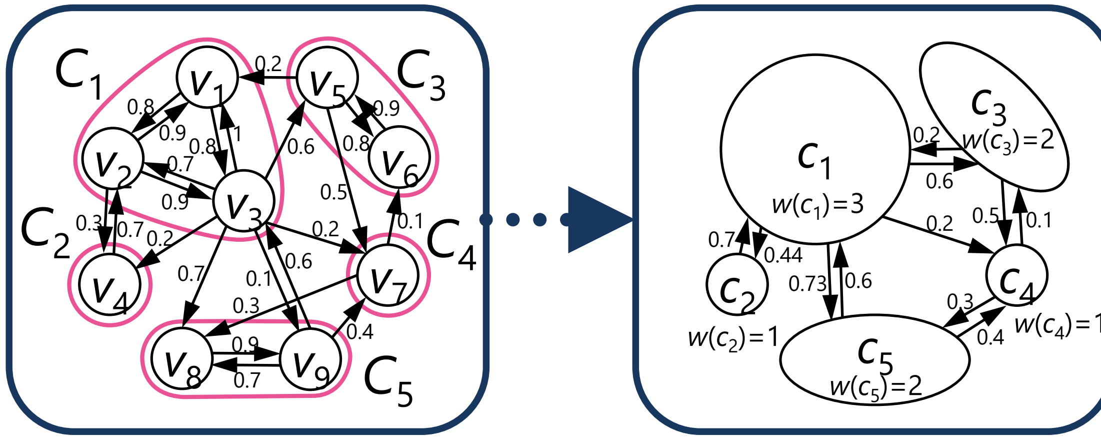
Conclusion

Scalable influence analysis through graph reduction

① Strategy

② Algorithm

③ Frameworks



Future directions

- Finding *better* vertex partitions
- Other reduction strategies
  Not so effective for the weighted probability (see our paper)
- Parallelization & dynamic updates (see our paper)