Coarsening Massive Influence Networks

for Scalable Diffusion Analysis

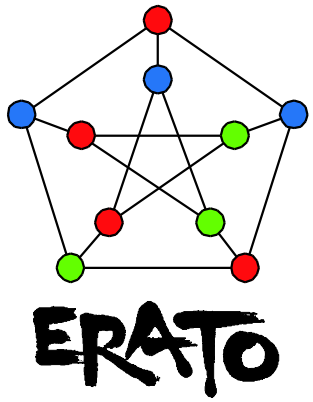
Naoto Ohsaka, Tomohiro Sonobe, Sumio Fujita, and Ken-ichi Kawarabayashi. Coarsening Massive Influence Networks for Scalable Diffusion Analysis. In SIGMOD, pages 635--650, 2017.

大坂直人 (東京大学)

薗部知大 (NII)

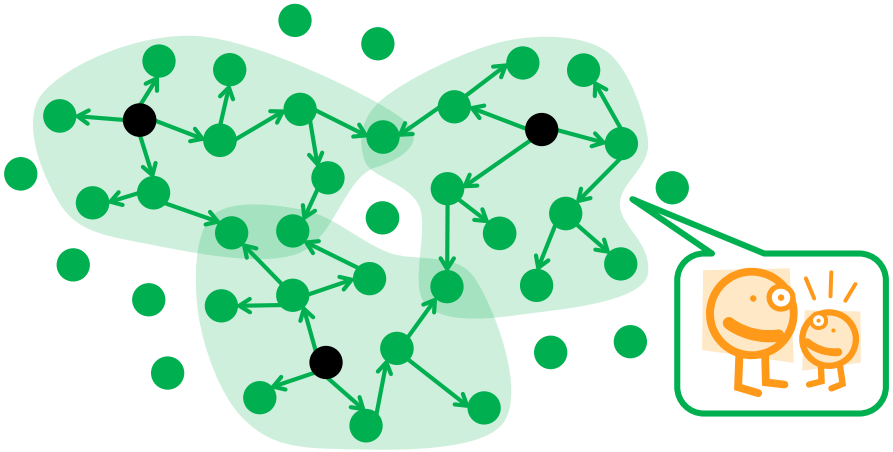
藤田澄男 (ヤフー株式会社 Yahoo! JAPAN 研究所)

河原林健一 (NII)



ソーシャルネットワーク上の情報拡散

情報拡散の単純な表現形式



Q. 最も影響力の高い集団をどのように発見するか?

マーケット戦略 [Domingos-Richardson. *KDD'01*]

II

影響最大化 (Influence maximization)

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

辺確率グラフ上のアルゴリズム

大規模なグラフにおける情報拡散解析

様々な影響最大化手法

Greedy [*KDD'03*], Static Greedy [*CIKM'13*], RIS [*SODA'14*], PMC [*AAAI'14*], TIM+ [*SIGMOD'14*], IMM [*SIGMOD'15*], Stop-and-Stare [*SIGMOD'16*] etc...

しかし...

絶対的な王者不在[Arora-Galhotra-Ranu. *SIGMOD'17*]

我々のゴール:

グラフの簡約化によるスケーラブルな情報拡散解析

グラフ簡約化の関連研究

特定の性質を保ちながらサイズを削減する手法

到達可能性 [Zhou-Zhou-Yu-Wei-Chen-Tang. *SIGMOD'17*]
クラスタリングの結果 [Satuluri-Parthasarathy-Ruan. *SIGMOD'11*]
uncertain graphの頂点次数 [Parchas-Gullo-Papadias-Bonchi. *SIGMOD'14*]
etc..

☹ 情報拡散の精度を保つことが目的ではない

辺確率グラフに対する簡約化手法

SPINE [Mathioudakis-Bonchi-Castillo-Gionis-Ukkonen. *KDD'11*]

COARSENET [Purohit-Prakash-Kang-Zhang-Subrahmanian. *KDD'14*]

☹ スケール性が低い

我々の貢献

簡略化戦略

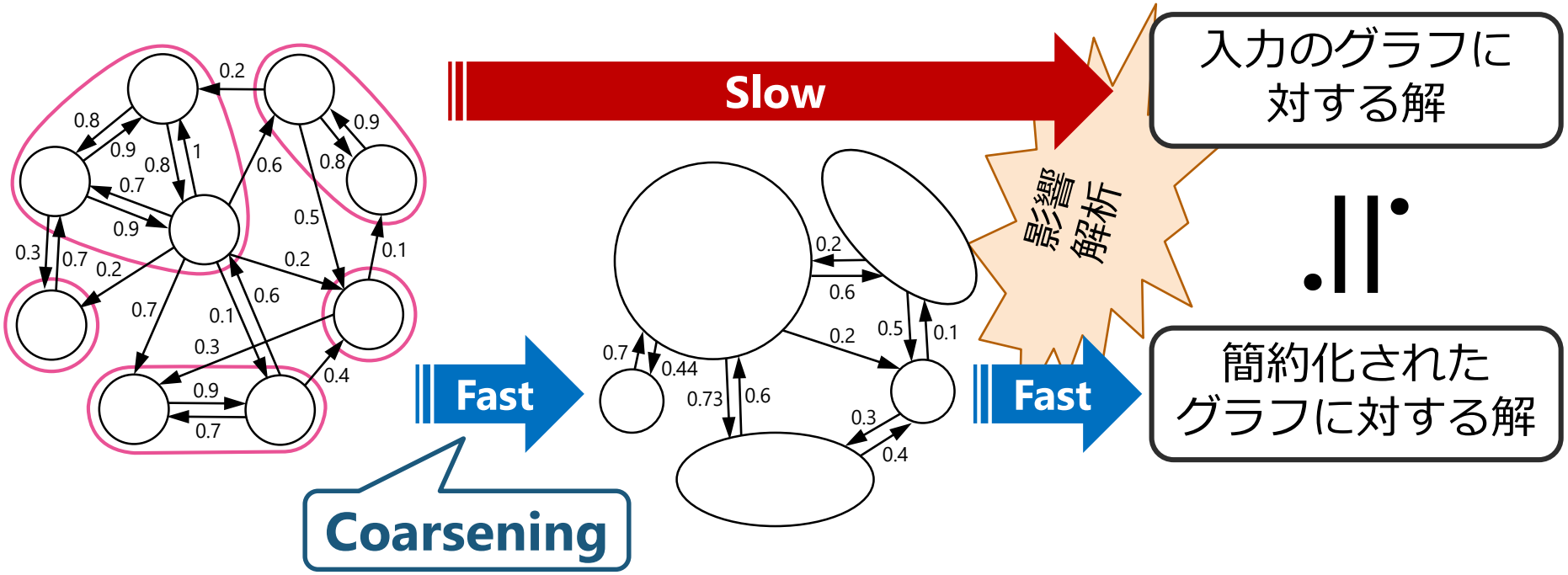
精度保証付

スケーラブルなアルゴリズム

10億エッジ/1時間

解析フレームワーク

2－30倍の高速化



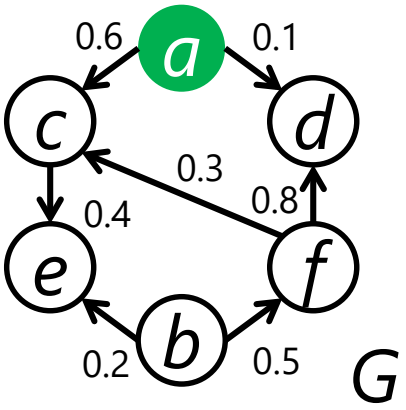
予備知識:

独立カスケードモデル

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

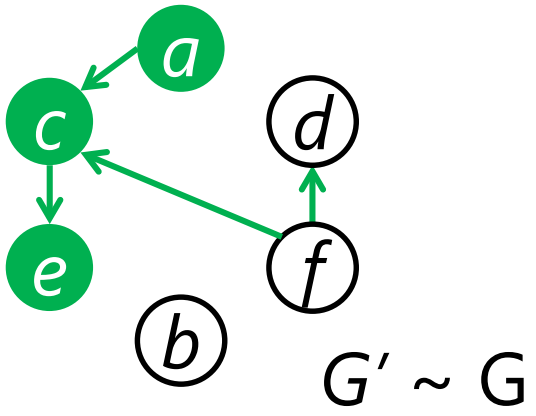
辺確率グラフ $G = (V, E, p)$ & Seed set $S \subseteq V$

Gに対する影響拡散	=	ランダムグラフ$G' \sim G$ 上での到達性



G

辺eは確率p_eで残す



$G' \sim G$

影響拡散 $\mathrm{Inf}_G(S)$ = $E_{G' \sim g}$[G'上でSから到達可能な頂点数]

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

予備知識:

二つの影響解析問題

影響力推定

入力: シードセットS

出力: $\mathrm{Inf}_G(S)$

− #P-hard [Chen-Wang-Wang. *KDD'10*]

＋ モンテカルロ法で近似が可能
ランダムグラフの生成を繰り返す

影響最大化

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

入力: 整数k

出力: $\mathrm{argmax}\ \mathrm{Inf}_G(S)$

S: $|S|=k$

− NP-hard [Kempe+*'03*]

＋ 貪欲法で$(1 - e^{-1}) \fallingdotseq 0.63$近似

$\mathrm{Inf}_G(\bullet)$は劣モジュラ [Kempe+*'03*]

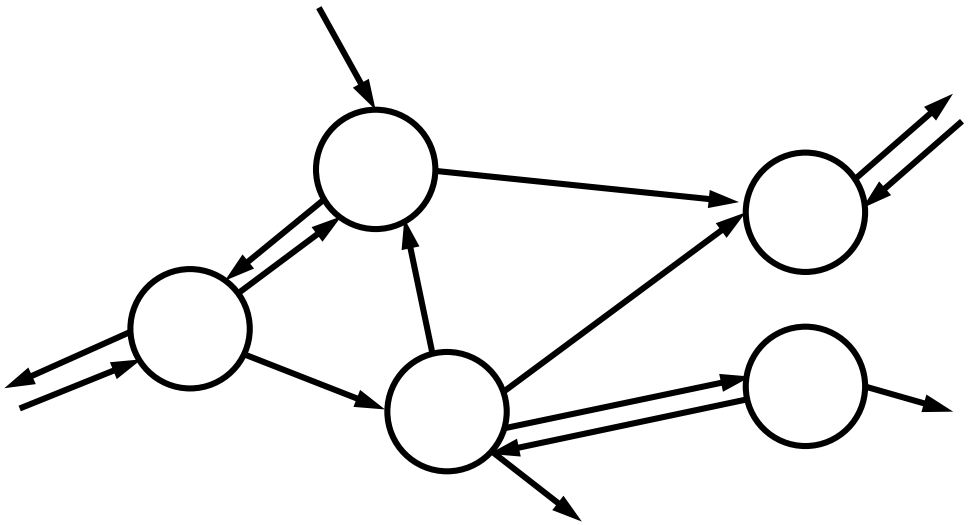
計算コスト ≒ 全辺をスキャンするコスト

提案戦略:

Design concept (1)

中心的アイディア = **Coarsening (粗大化)**

特定の集合内の頂点を互いに区別しない



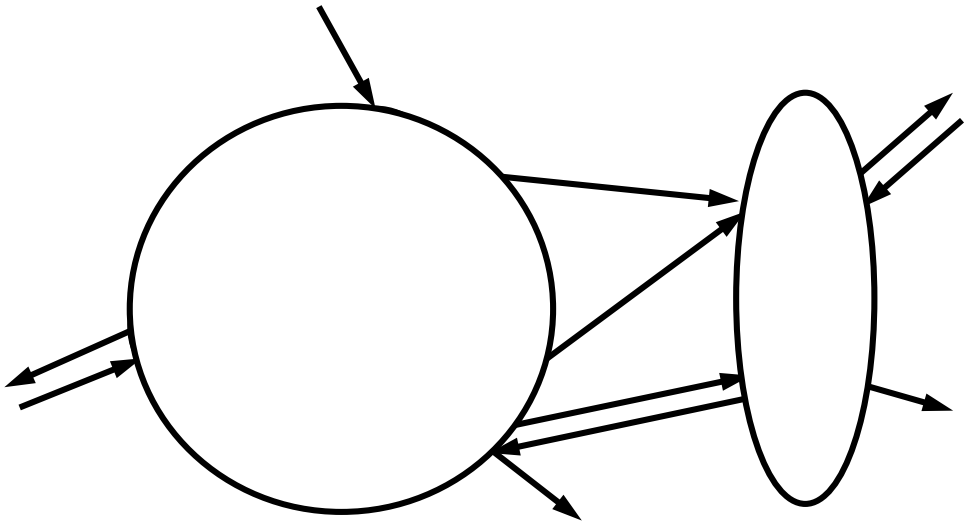
+ 辺数を大きく減らすことが可能

提案戦略:

Design concept (2)

中心的アイディア = **Coarsening (粗大化)**

特定の集合内の頂点を互いに区別しない



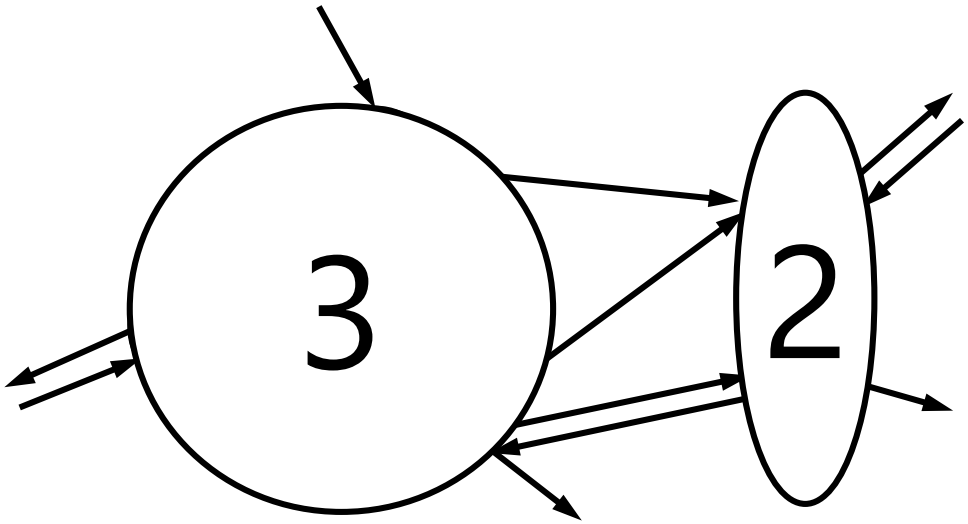
+ 辺数を大きく減らすことが可能

提案戦略:

Design concept (3)

中心的アイディア = **Coarsening (粗大化)**

特定の集合内の頂点を互いに区別しない



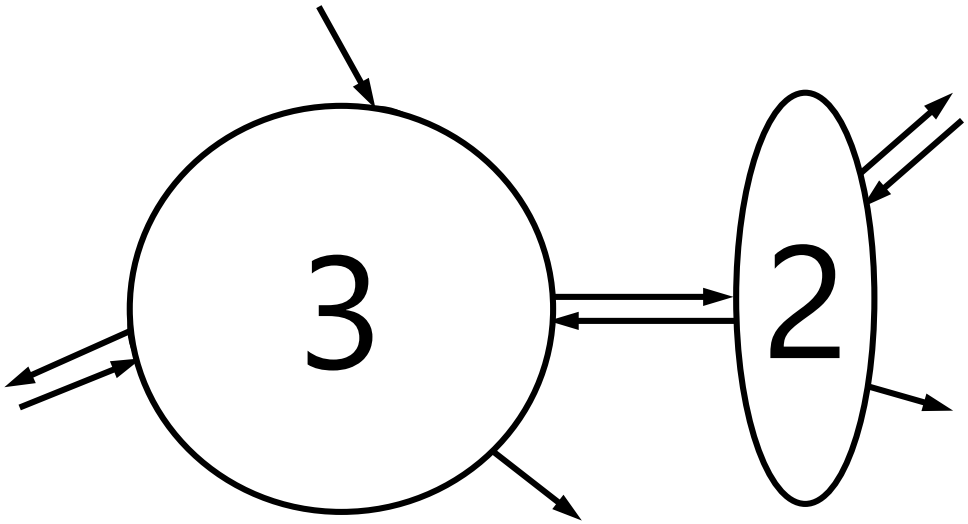
+ 辺数を大きく減らすことが可能

提案戦略:

Design concept (4)

中心的アイディア = **Coarsening (粗大化)**

特定の集合内の頂点を互いに区別しない



+ 辺数を大きく減らすことが可能

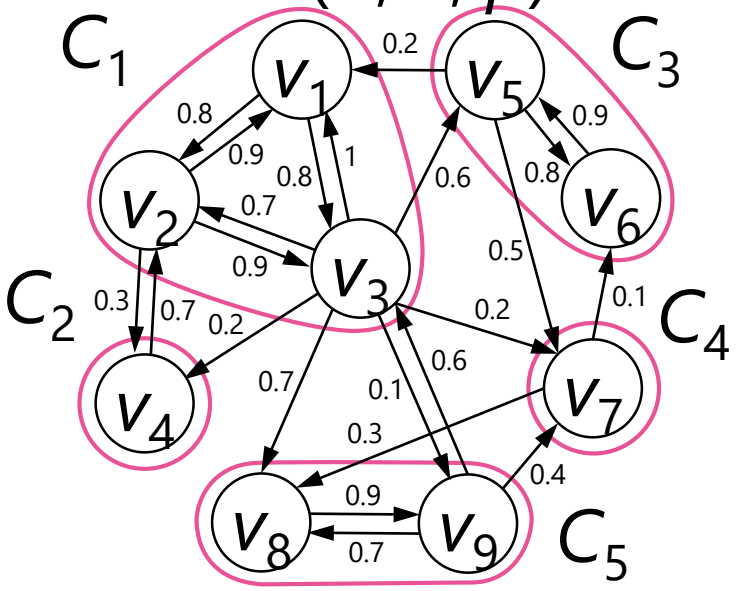
提案戦略:

粗大化した辺確率グラフ

頂点の分割$P = \{C_j\}_j$を与える

辺確率グラフ

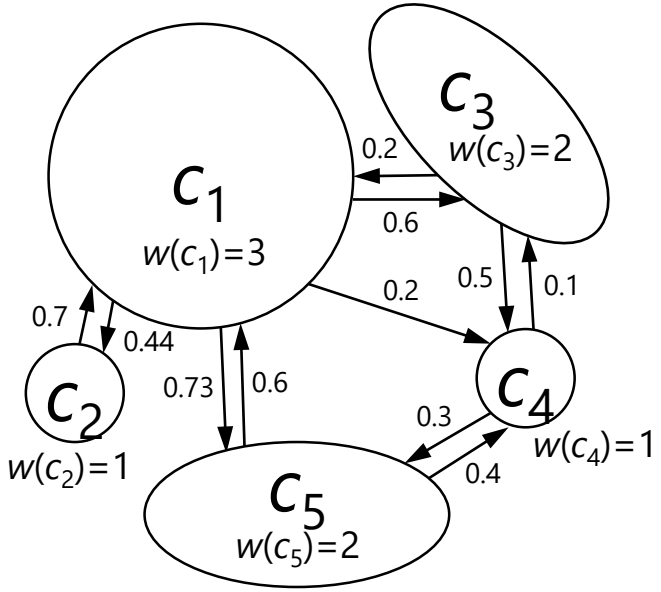
$G = (V, E, p)$



Pに関して粗大化

粗大化したグラフ

$H = (W, F, q)$ & 重みw



提案戦略:

粗大化した辺確率グラフ

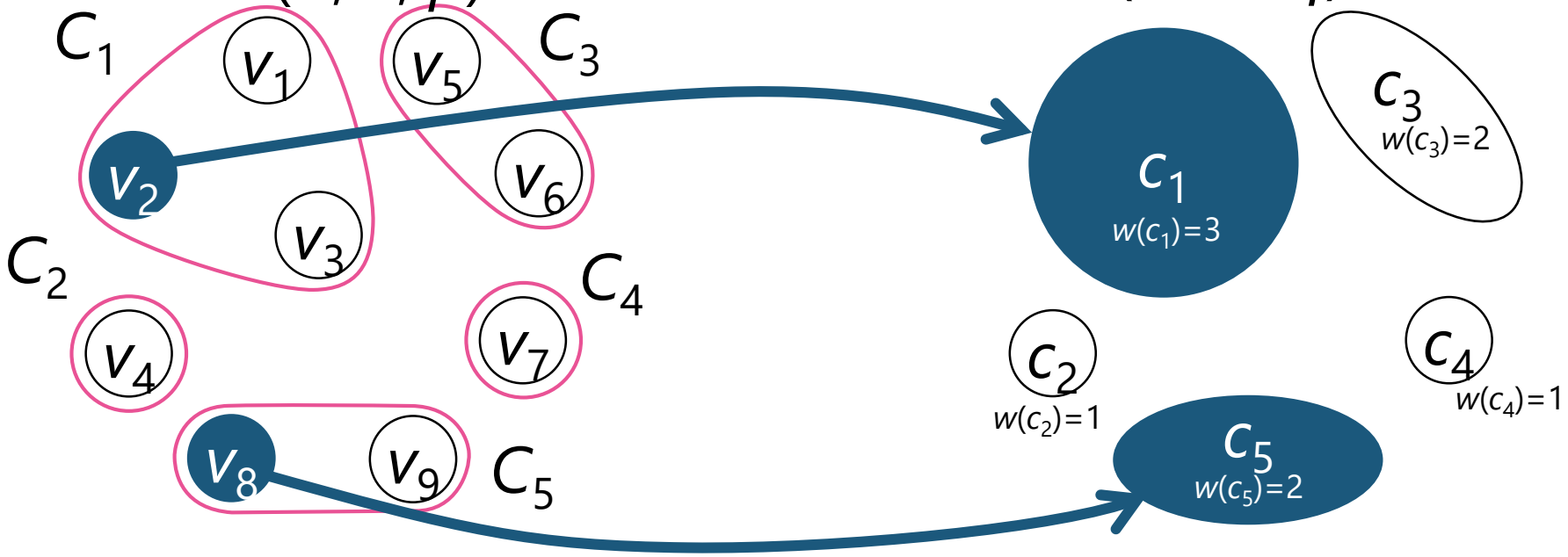
頂点の分割$P = \{C_j\}_j$を与える

辺確率グラフ

$G = (V, E, p)$

粗大化したグラフ

$H = (W, F, q)$ & 重みw



C_jの中の頂点 ⇒ 重み付き頂点c_j

$|C_j| = w(c_j)$

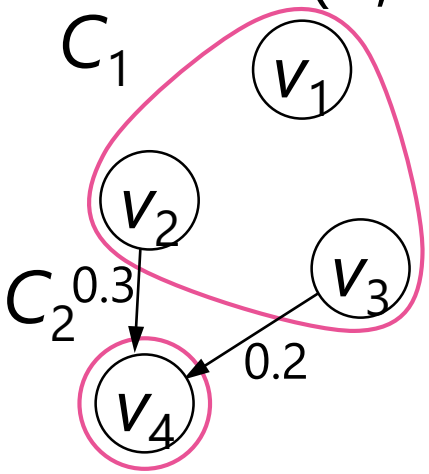
提案戦略:

粗大化した辺確率グラフ

頂点の分割$P = \{C_j\}_j$を与える

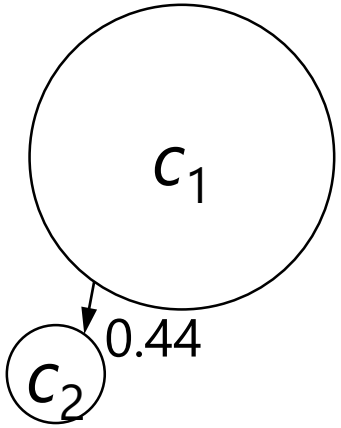
辺確率グラフ

$G = (V, E, p)$



粗大化したグラフ

$H = (W, F, q)$ & 重みw



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

$$1 - (1 - p_{v2v4})(1 - p_{v3v4}) = q_{c1c2}$$

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

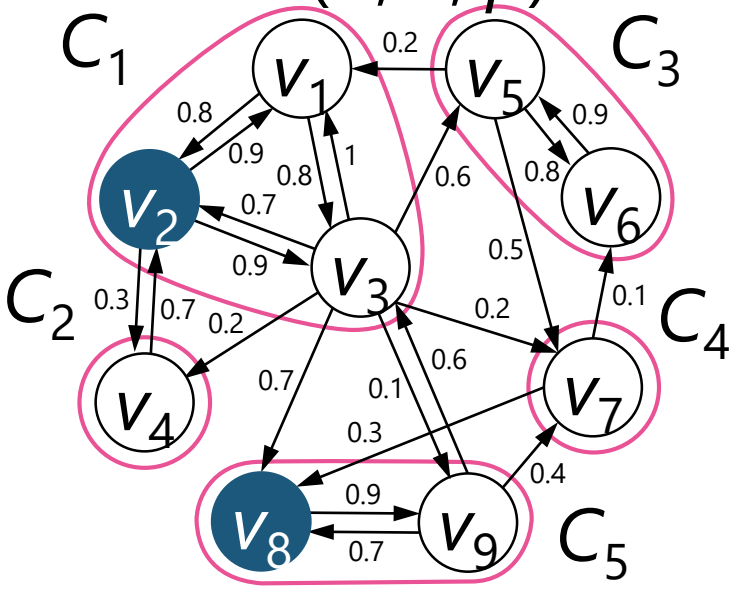
提案戦略:

粗大化した辺確率グラフ

頂点の分割$P = \{C_j\}_j$を与える

辺確率グラフ

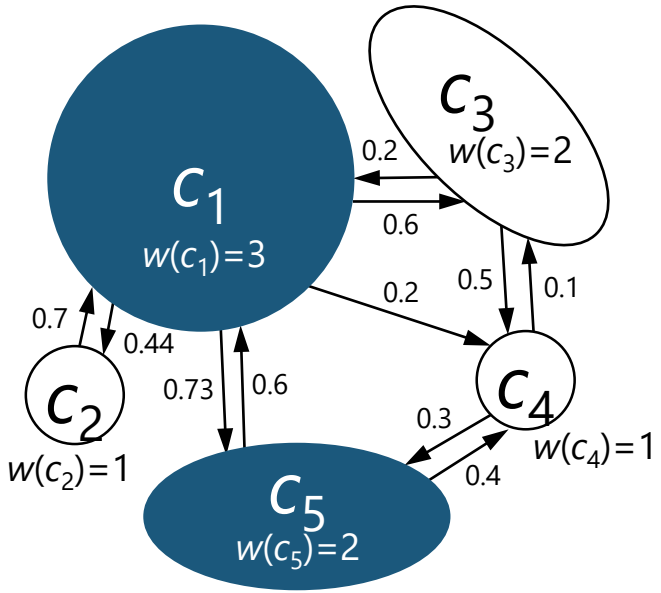
$G = (V, E, p)$



Pに関して粗大化

粗大化したグラフ

$H = (W, F, q)$ & 重みw



望ましい性質: $\mathrm{Inf}_G(\{v_2, v_8\}) \fallingdotseq \mathrm{Inf}_H(\{c_1, c_5\})$

では良い分割とは?

提案戦略

GとHの影響力の差

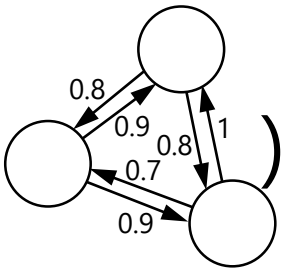
Theorem 4.6

小さいほど良い

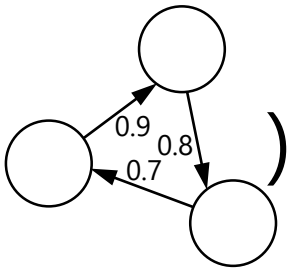
$$\mathrm{Inf}_G(\bullet) \leqq \mathrm{Inf}_H(\bullet) \leqq \frac{1}{\prod_{C_j \in P} \mathrm{Rel}(g[C_j])} \cdot \mathrm{Inf}_G(\bullet)$$

$\mathrm{Rel}(G) := \Pr_{G' \sim G}[G'$が**強連結**$]$ (“Rel”iability)

$G[C_j] := C_j$によって誘導されるGの部分グラフ

Rel (

) = 0.88848

Rel (

) = 0.504

提案戦略

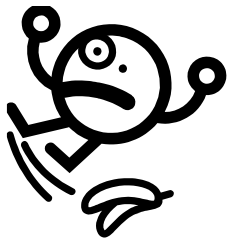
良い分割とは

Π Rel($g[C_j]$)が高くなるような分割Pが欲しい

- Rel(•)の厳密計算は #P-hard

  [Valiant. SIAM J. Comput.'79] [Ball. Networks'80]

- 近似計算でも多くのサンプリング(ランダムグラフの生成)が必要



高い値のRel(•)を備える頂点集合がほしいが
少ないサンプリングでどう実現するか?

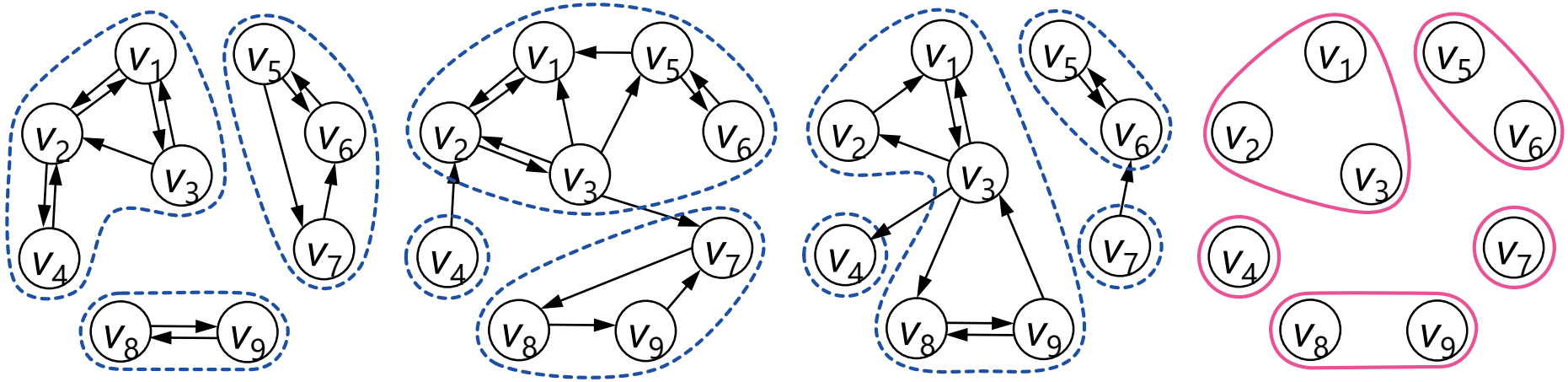
r-robust SCC strongly connected components

少数(*r*個)のランダムグラフで常に強連結な頂点集合は、
他のランダムグラフでも強連結だろう

提案戦略

r-robust SCCの定義

頂点集合*C*が*r*個のランダムグラフG_1, G_2, …, G_rに関して*r*-robust SCCであるとは、
①全てのG_iで*C*が強連結
②*C*は極大



部分グラフG_1　部分グラフG_2　部分グラフG_3　*3*-robust SCCs

*G*から各辺eを確率p_eでサンプリングして生成

提案戦略

r-robust SCCの制限と利点

$\prod_{C_j \in P} \mathrm{Rel}(g[C_j])$に理論保証無し

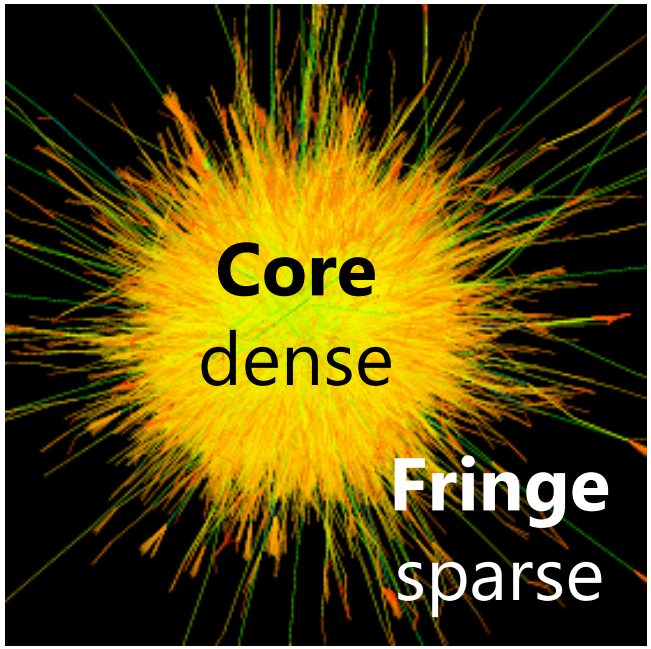
P:= *r*-robust SCCの集合

理論面からの妥当性

Theorem 4.12
r-robust SCCは高いRel(•)を備える頂点集合
-> Inf(•)が保たれることが期待

Theorem 4.13
r-robust SCCは密
-> 辺数を大幅に削除可能

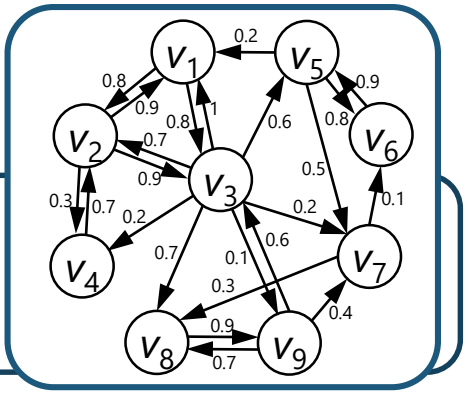
Core-fringe structure
[Leskovec-Lang-Dasgupta-Mahoney. *WWW'08*]
[Maehara-Akiba-Iwata-Kawarabayashi. *PVLDB'14*]



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

アルゴリズム

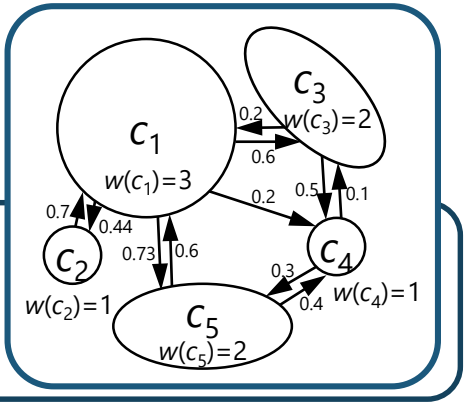
入力: $G = (V, E, p)$ & r



Stage 1: r-robust SCCの抽出(並列化可能)

Stage 2: 各r-robust SCCを粗大化

出力: $H = (W, F, q)$ & w

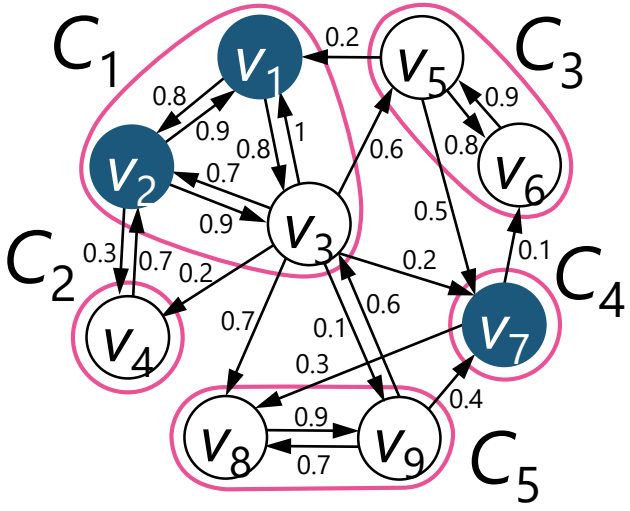


Speed-oriented	O(r(|V|+|E|)) time	O(r(|V|+|E|))space
Scalability-oriented Disk-based SCC algorithms Space reduction technique	O(r(|V|+|E|)+t_A(V, E)) time	O(r(|V|+|F'|)) space |F'| << |F| in practice

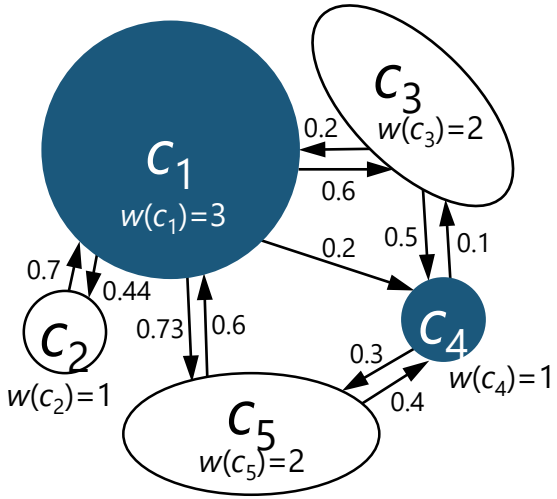
提案フレームワーク

影響力推定

計算対象: $\mathrm{Inf}_G(S)$



1. SをHにマップ



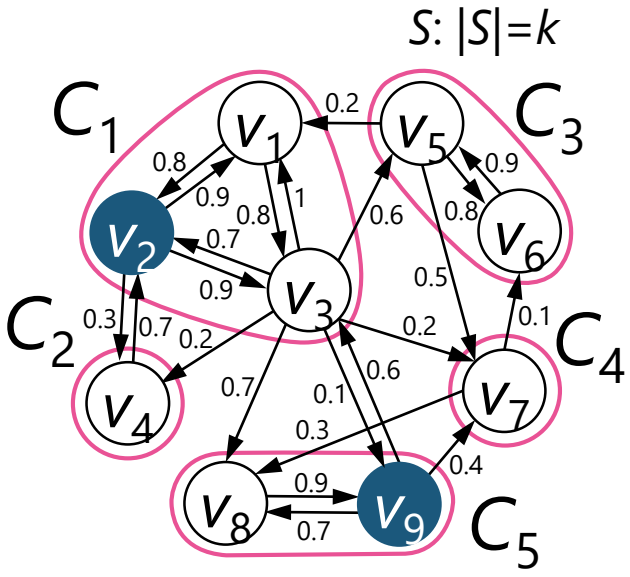
2. $\mathrm{Inf}_H(T)$を
既存手法で推定

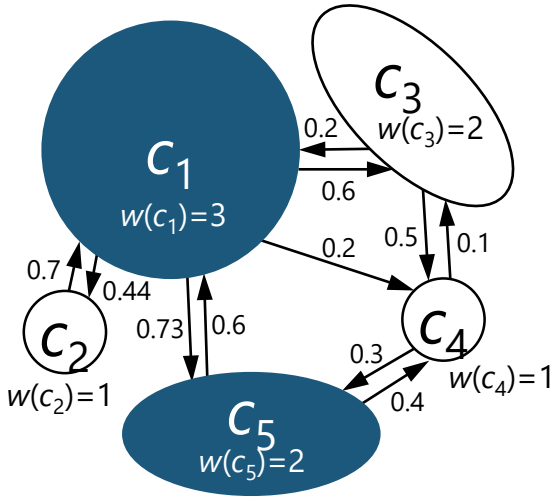
$\mathrm{Inf}_G(\{ v_1\ v_2\ v_7 \})$ ≒ $\mathrm{Inf}_H(\{ c_1\ c_4 \})$

提案フレームワーク

影響最大化

計算対象: argmax $\mathrm{Inf}_G(S)$





1. 既存手法でhから
大きさkのTを抽出

S = ({ v_2 v_9 })

T = ({ c_1 c_5 })

2. TをGにマップ

実験

設定

ソーシャル、コミュニケーション、ウェブグラフ

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

確率設定

- exponential 平均0.1の指数分布 [Dickens+'12] から考案
- trivalency 0.1, 0.01, 0.001 [Chen+'10]
- weighted 入次数の逆数 [Kempe+'03]
- uniform 0.1で統一 [Kempe+'03]

アルゴリズム

- *r*=16 (デフォルト)
- Disk-based SCC algorithm [Laura-Santaroni. *TAPAS'11*]

実験環境

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

実験

データセット

name	type	|V|	|E|
ca-HepPh	collab.(u)	9,877	51,946
soc-Slashdot0922	social	82,168	870,161
web-NotreDame	web	325,729	1,469,679
wiki-Talk	commu.	2,394,385	5,021,410
com-Youtube	social(u)	1,134,890	5,975,248
higs-twitter	social	456,626	14,855,819
soc-Pokec	social	1,632,803	30,622,564
soc-LiveJournal1	social	4,847,571	68,475,391
com-Orkut	social(u)	3,072,441	234,370,166
twitter-2010	social	41,652,230	1,468,364,884
com-Frinedster	social(u)	65,608,366	3,612,134,270
uk-2007-05	web	105,218,569	3,717,169,969
ameblo	web	272,687,914	6,910,266,107

実験

実行時間 & メモリ使用量

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

時間&空間計算量ともに大規模なグラフまでスケール

10× slower

10× smaller

実験

並列化

dataset	|V|	|E|	speed-oriented T=1	speed-oriented T=16	scalability-oriented T=1	scalability-oriented T=16
soc-Slashdot0922	0.1M	0.9M	< 1 s	< 1 s	6 s	2 s
wiki-Talk	2M	5M	42 s	12 s	57 s	22 s
soc-Pokec	2M	31M	35 s	11 s	224 s	52 s
soc-LiveJournal1	5M	68M	95 s	33 s	508 s	127 s
twitter-2010	42M	1,468M	1,763 s	612 s	11,522 s	2,620 s
com-Friendster	66M	3,612M	3,964 s	1,143 s	26,424 s	5,844 s
uk-2007-05	105M	3,717M	3,106 s	1,036 s	29,540 s	6,329 s
ameblo	273M	6,910M	—	—	35,761 s	7,613 s

約3倍　　約5倍

実験

グラフサイズの縮小率

$G = (V, E, p)$ 入力: 元のグラフ

$H = (W, F, q)$ 出力: 粗大化したグラフ

dataset	$\|V\|$	$\|E\|$	$\|W\| / \|V\|$ ≫	$\|F\| / \|E\|$
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%

辺数の大幅な削減に成功

巨大で密なr-robust SCCが存在

実験

影響力推定

手法: モンテカルロシミュレーション

ランダムに選択した頂点からの拡散を10000回調査

dataset	run time			
	Monte-Carlo	Our framework w/ *Monte-Carlo*	time reduction	≈ edge 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%

ほぼ精度を保ちながら高速に推定(平均絶対誤差≦0.1、順位相関係数≧0.86)

実験

影響最大化

手法: *D-SSA* [Nguyen-Thai-Dinh. *SIGMOD'16*]

k=100の頂点集合を抽出

dataset		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	—

解の精度もほぼ保たれている(誤差1%未満)

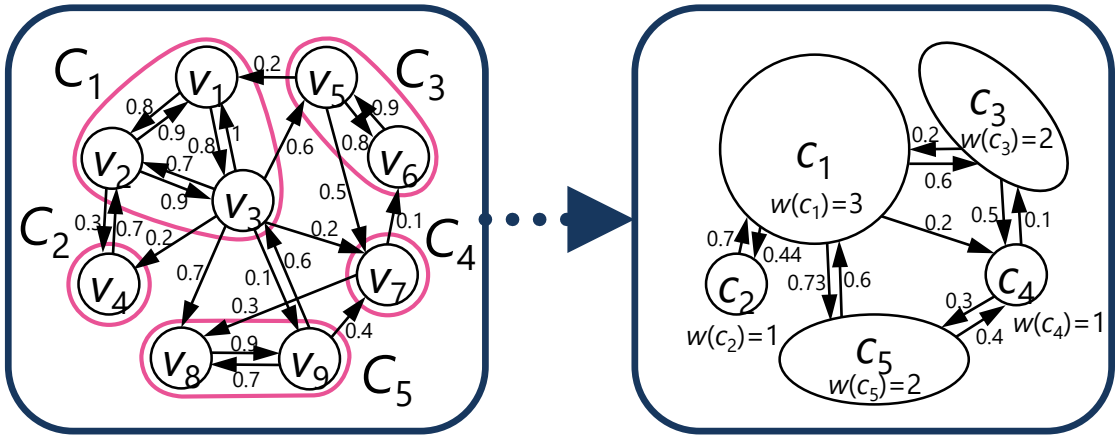
結論

グラフ簡約化を用いたスケーラブルな影響解析

① 戦略

② アルゴリズム

③ フレームワーク



今後

- より良い頂点分割の方法
- 動的なグラフへの対応