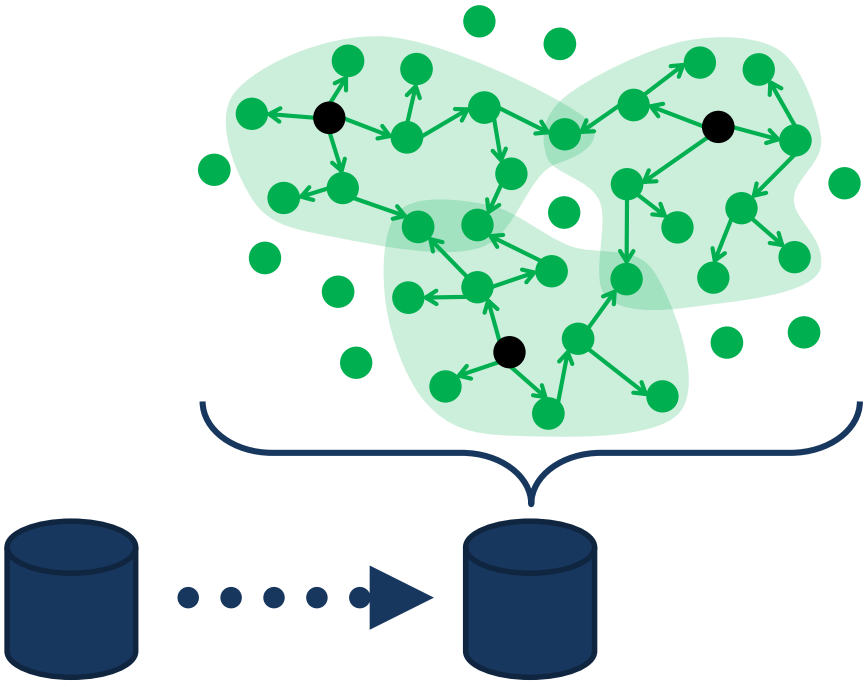
Dynamic Influence Analysis in Evolving Networks

42nd International Conference on Very Large Data Bases
Proceedings of the VLDB Endowment, 9(12):1077–1088, 2016

大坂 直人 (東京大学)
秋葉 拓哉 (PFN)
吉田 悠一 (NII & PFI)
河原林 健一 (NII)



42nd International Conference on Very Large Data Bases

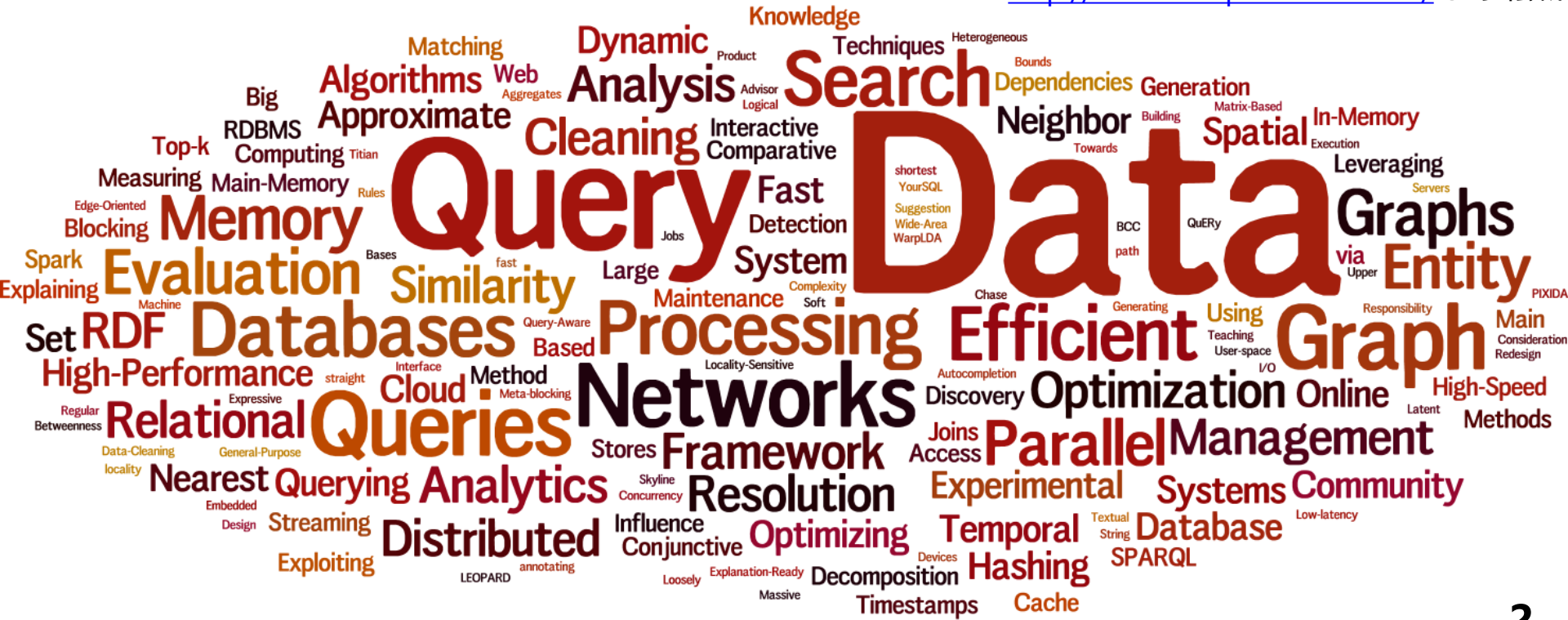
本部がインド

Organizing Sponsor

PERSISTENT

- 2016/9/5～9/9 @ ニューデリー，インド
- 採択論文数: 104
  ※41stの採択率＝20%

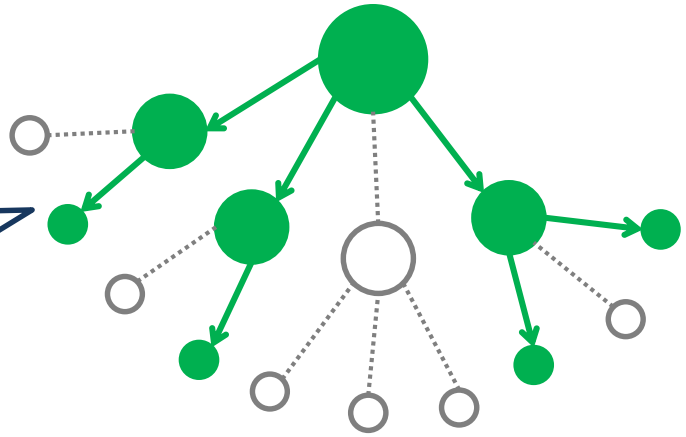
http://vldb2016.persistent.com/ より転載



利用サービス：Wordle http://www.wordle.net/

ネットワーク上の拡散

噂・口コミ
意見
感染症
イノベーション
汚染



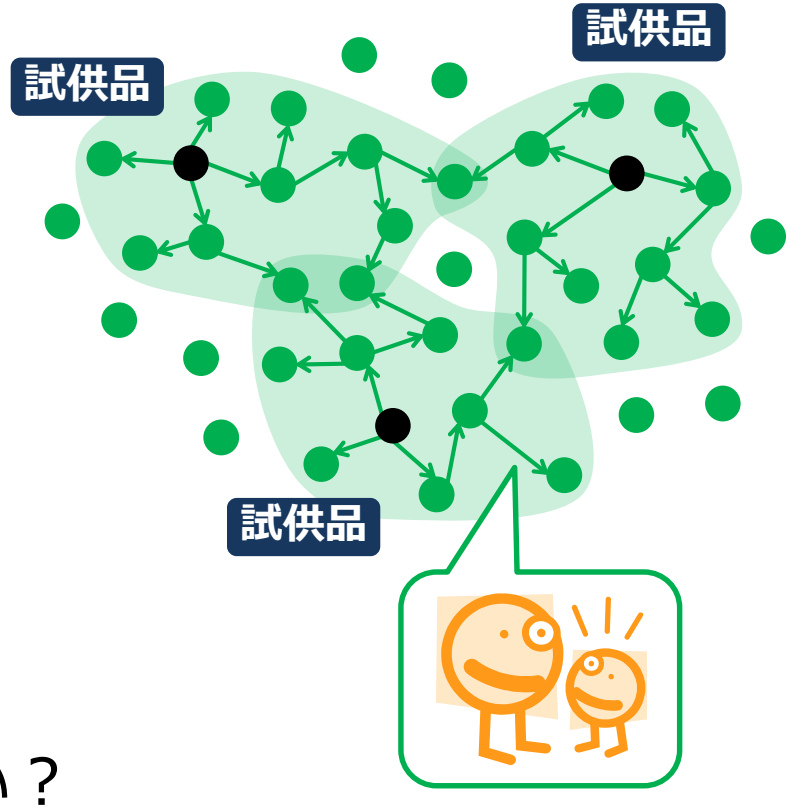
2000年～ オンラインソーシャルネットワークの台頭
膨大な個人単位の履歴が直ぐに手に入る

人々の過程と活動を理解・予測・制御したい！

モデル化	[Rodriguez-Balduzzi-Schölkopf. *ICML'11*]
パラメタ学習	[Goyal-Bonchi-Lakshmanan. *WSDM'10*]
ネットワーク推定	[Rodriguez-Leskovec-Krause. *KDD'10*]
拡散の将来予測	[Cheng-Adamic-Dow-Kleinberg-Leskovec. *WWW'14*]
デマ拡大防止	[Budak-Agrawal-Abbadi. *WWW'11*]

我々: グラフアルゴリズム的問題に焦点

取り組む2つの問題



影響最大化

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

Q. 拡散が最も広がる集団はどれ？
バイラルマーケティングへの応用

[Domingos-Richardson. *KDD'01*]

影響力推定

Q. この集団の拡散力はどれくらい？

今回の話：

巨大・動的グラフ上の計算効率への挑戦

計算効率の観点での挑戦

使用データセット

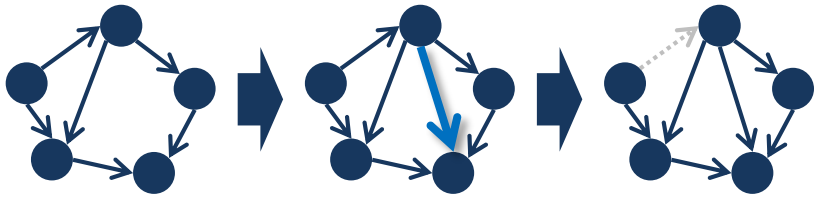
YouTube	320万点	1,880万辺
Flickr	230万点	3,310万辺

1. グラフが巨大

百万点超 ⇝ $\mathcal{O}$(**点数^2**)**時間**は✗

私の取組 [Ohsaka-Akiba-Yoshida-Kawarabayashi. *AAAI'14*] (感謝祭'14)

友達関係の成立・解消



2. グラフが動的・成長

最新の解析結果を追跡したい

静的手法の逐次適用 ⇝ **線形時間**以上

[Hayashi-Akiba-Yoshida. *VLDB'16*] 媒介中心性

[Ohsaka-Maehara-Kawarabayashi. *KDD'15*] PageRank (感謝祭'15)

[Akiba-Iwata-Yoshida. *WWW'14*] 最短経路クエリ (感謝祭'14)

これまでの状況

1. グラフが巨大

☺ およそ**解決**

ほぼ線形時間近似手法の登場

[Borgs-Brautbar-Chayes-Lucier. *SODA'14*]

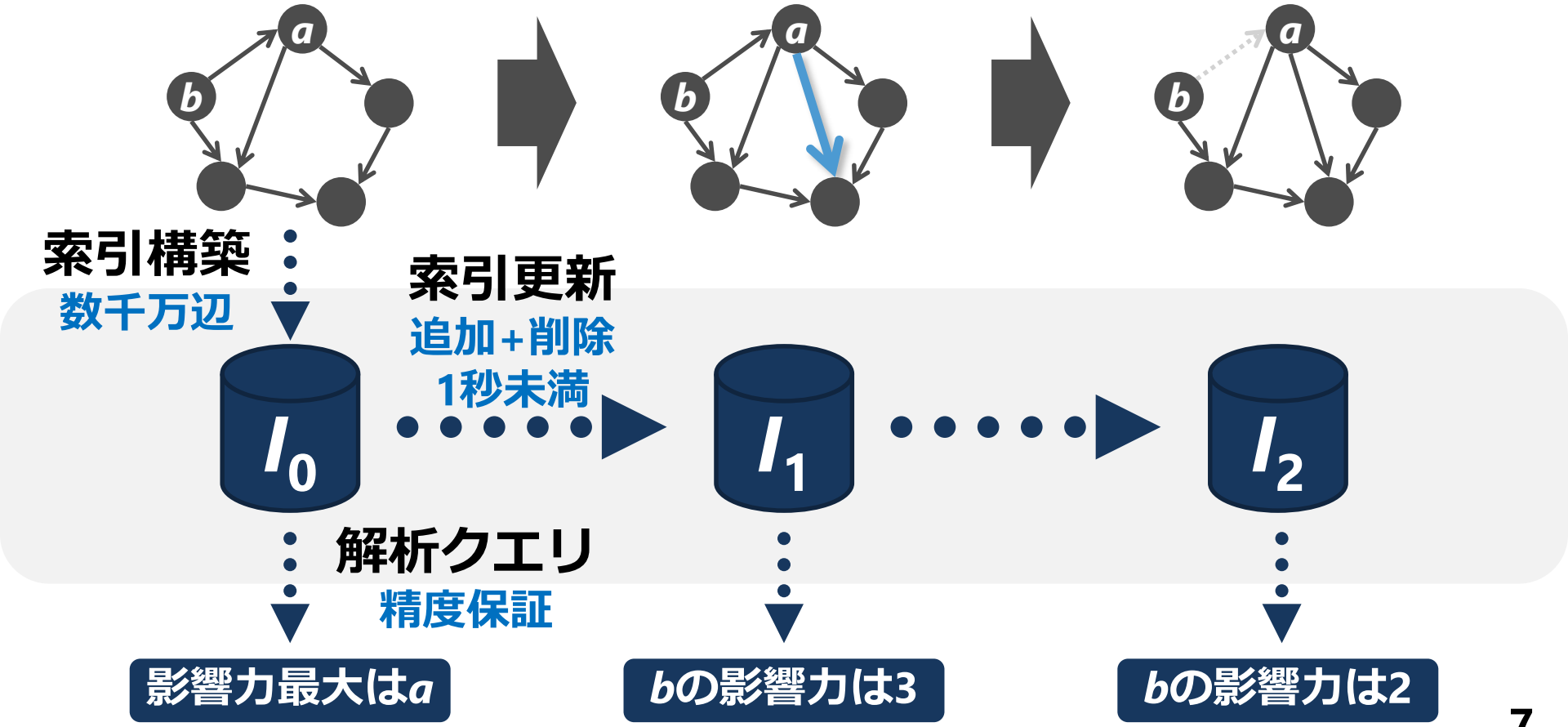
2. グラフが動的・成長

☹ ほぼ**未開拓**

[Zhuang-Sun-Tang-Zhang-Sun. *ICDM'13*] グラフ変化の検知
[Chen-Song-He-Xie. *SDM'15*] 限られた状況のみ

本研究の貢献

成長するグラフ上の影響解析をサポートする **完全動的索引手法**の提案



予備知識

問題定義と既存のアプローチ

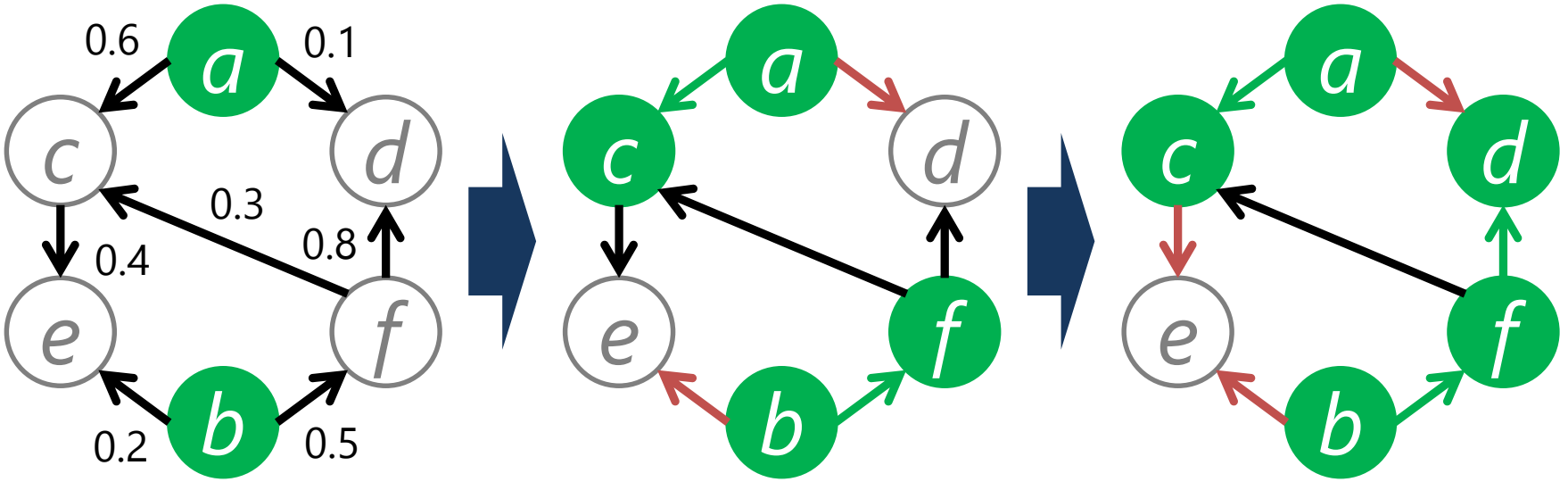
扱う拡散モデル 独立カスケード

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

辺確率つきグラフ $G = (V, E, p)$
シード集合 $S \subseteq V$

頂点の状態を初期化
- S内の頂点は**活性**
- S外の頂点は**非活性**

活性uから非活性v (一回きり)
- **成功** $w.p.$ p_{uv}
- **失敗** $w.p.$ $1 - p_{uv}$



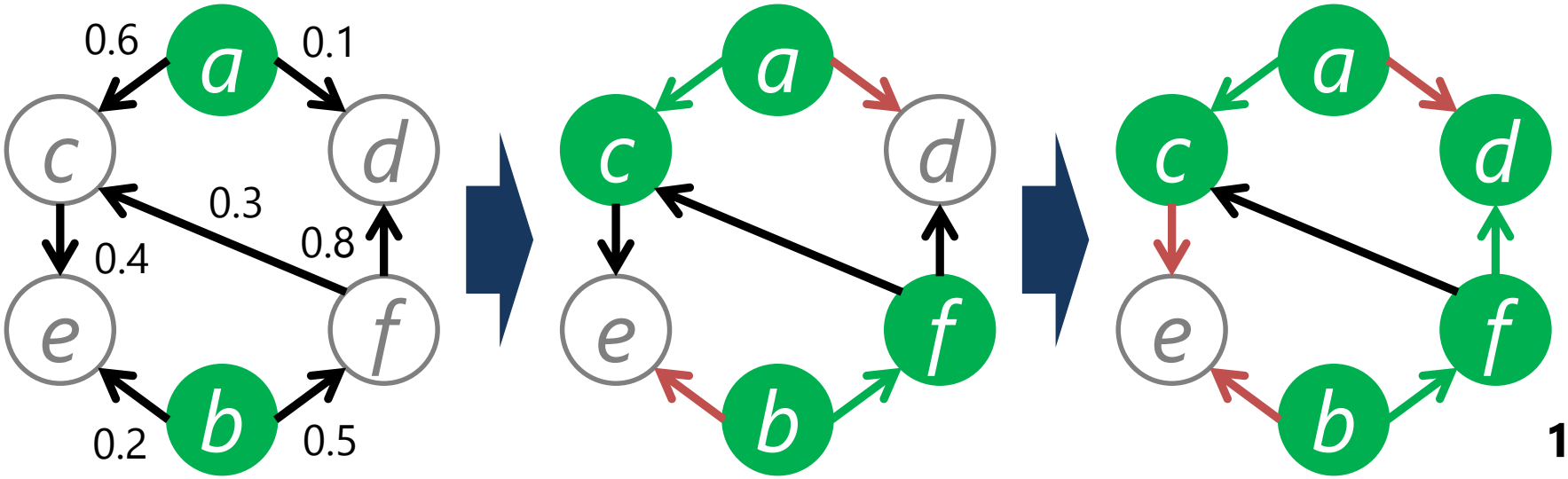
扱う拡散モデル 独立カスケード

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

辺確率つきグラフ $G = (V, E, p)$
シード集合 $S \subseteq V$

影響力

$$\sigma(S) := \mathbf{E}[S\text{がシード時の}\textbf{活性頂点}\text{数}]$$



問題定義

影響力推定

入力 頂点集合 S

出力 $\sigma(S)$

厳密計算は**#P-hard**

[Chen-Wang-Wang. *KDD'10*]

Monte-Carloで**良近似**

影響最大化

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

入力 整数 k

出力 $\underset{S:|S|=k}{\operatorname{argmax}}\ \sigma(S)$

厳密計算は**NP-hard** [Kempe+'03]

貪欲アルゴリズムで

$\left(1-\mathrm{e}^{-1}\right) \approx$ **63% 近似**

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

$\sigma(\cdot)$は**単調**・**劣モジュラ** [Kempe+'03]

劣モジュラ性 (限界効用逓減性)

$$\forall X \subseteq Y, v \notin Y, \sigma(X+v)-\sigma(X) \geq \sigma(Y+v)-\sigma(Y)$$

$\sigma(\cdot)$を高速・精確に評価したい！

スケッチ手法 **RIS**

[Borgs-Brautbar-Chayes-Lucier. *SODA'14*]

繰り返し：
- **ターゲット**頂点 z を無作為に選択
- **スケッチ**＝(zに影響する頂点集合)

逆向き
シミュレーション

スケッチ手法 **RIS**

[Borgs-Brautbar-Chayes-Lucier. *SODA'14*]

繰り返し：
- **ターゲット**頂点 z を無作為に選択
- **スケッチ**＝(zに影響する頂点集合)

逆向き
シミュレーション

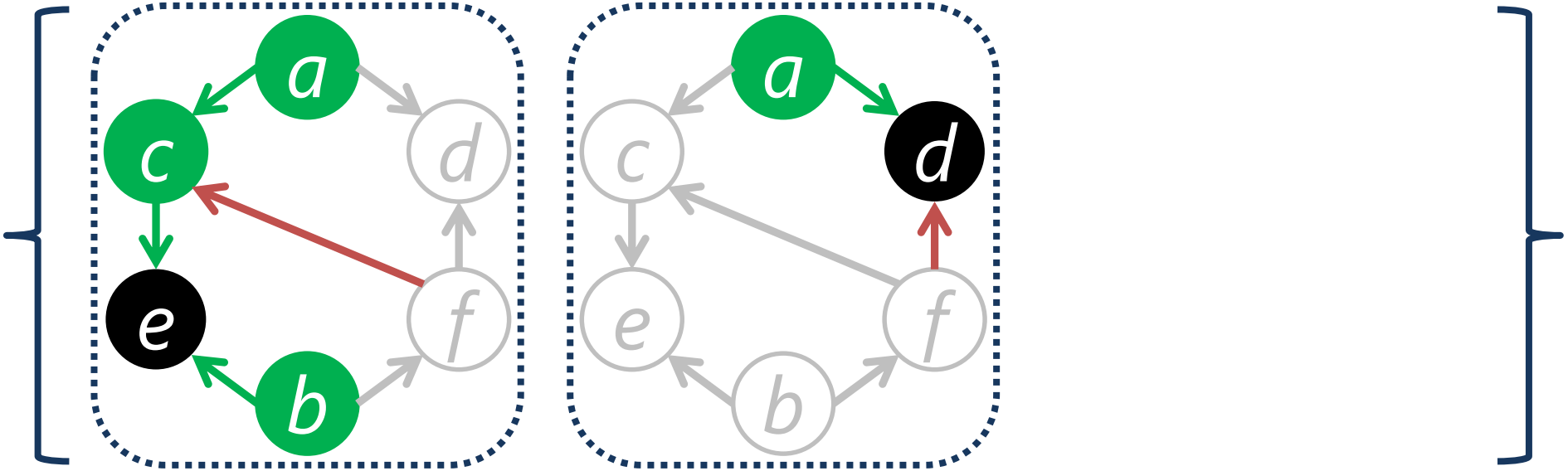
スケッチ手法 **RIS**

[Borgs-Brautbar-Chayes-Lucier. *SODA'14*]

繰り返し：

- **ターゲット**頂点 z を無作為に選択
- **スケッチ**＝(zに影響する頂点集合)

逆向き
シミュレーション

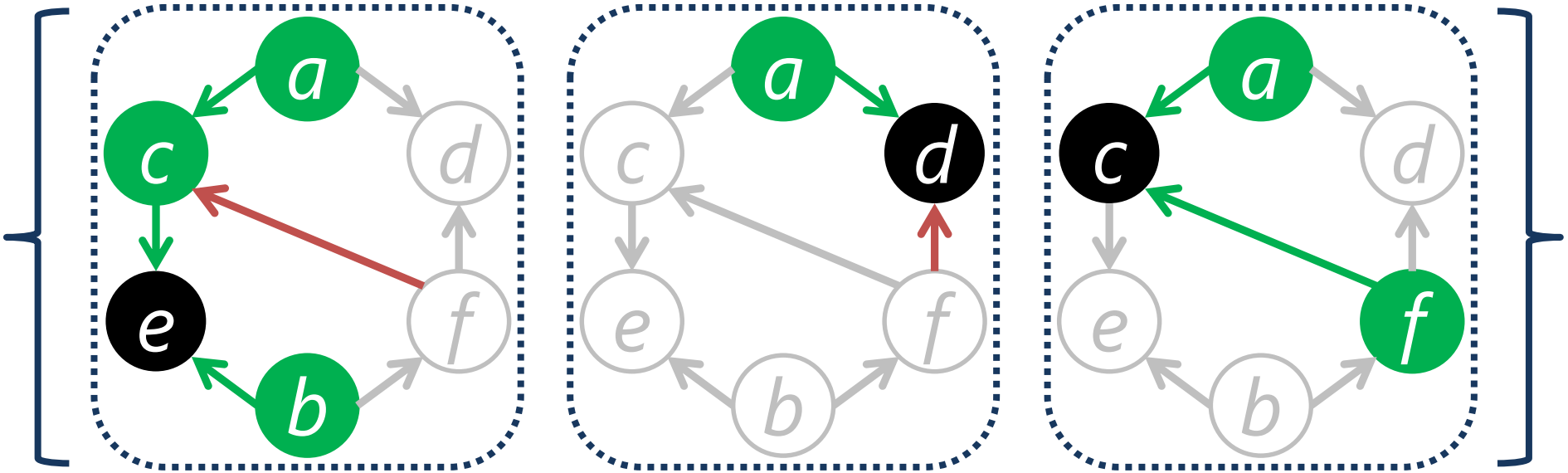


スケッチ手法 **RIS**

[Borgs-Brautbar-Chayes-Lucier. *SODA'14*]

繰り返し：
- **ターゲット**頂点 z を無作為に選択
- **スケッチ**＝(zに影響する頂点集合)

逆向き
シミュレーション

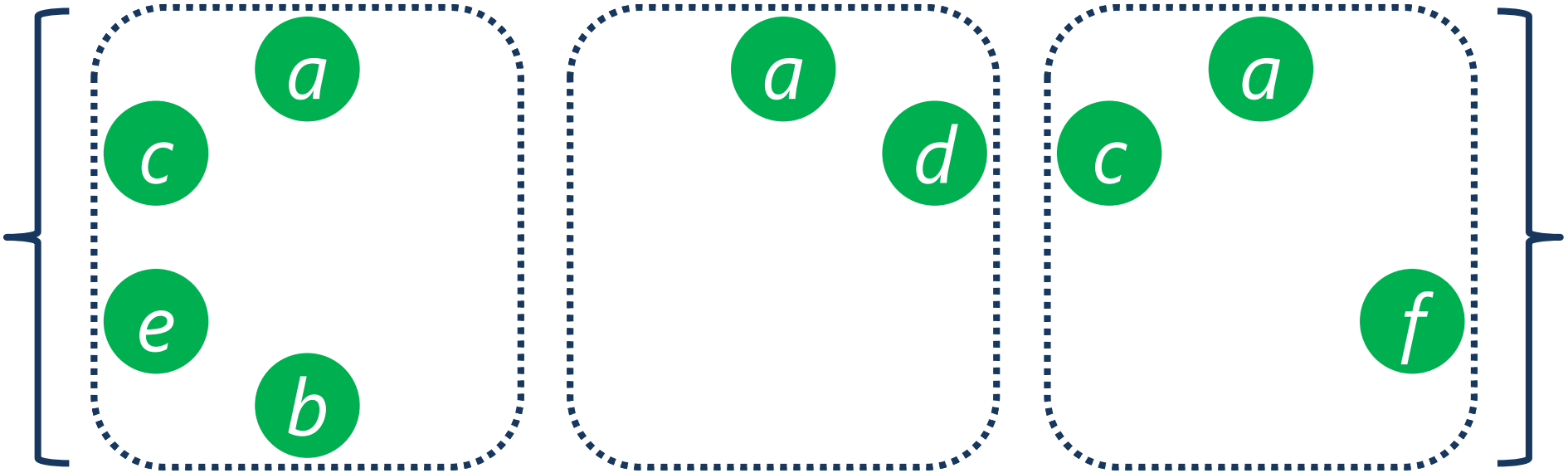


スケッチ手法 **RIS**

[Borgs-Brautbar-Chayes-Lucier. *SODA'14*]

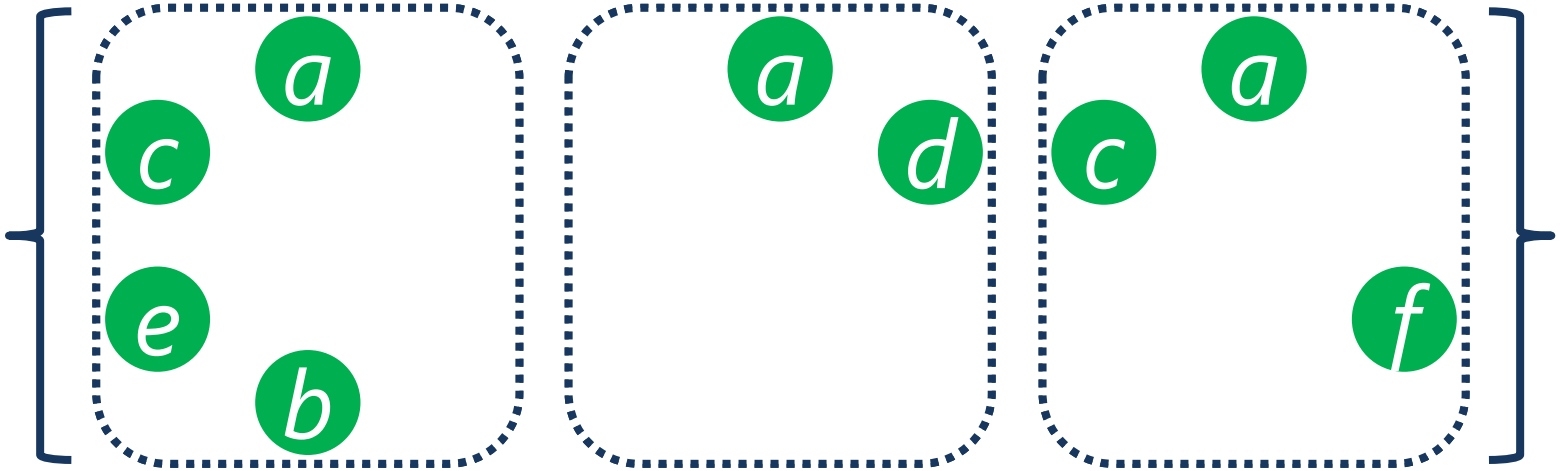
繰り返し：
- **ターゲット**頂点 z を無作為に選択
- **スケッチ**＝(zに影響する頂点集合)

逆向き
シミュレーション



スケッチ手法 **RIS**

[Borgs-Brautbar-Chayes-Lucier. *SODA'14*]



スケッチに多く現れる頂点は影響力が高そう

$\sigma(S) \propto \mathbf{E}[S$と交差するスケッチ数$]$

影響力推定 ⇝ Unionのサイズ

影響最大化 ⇝ Maximum Coverage

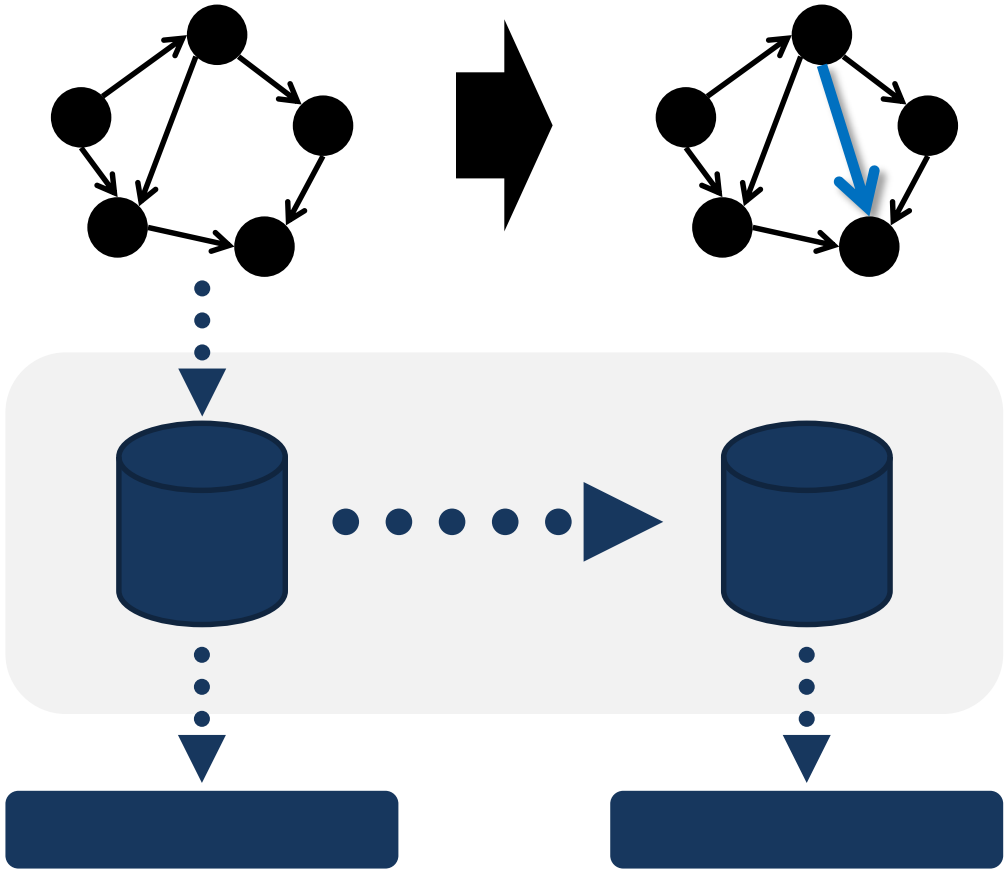
ほぼ線形サイズのスケッチで十分

我々の目標＝このスケッチを動的に更新

提案手法

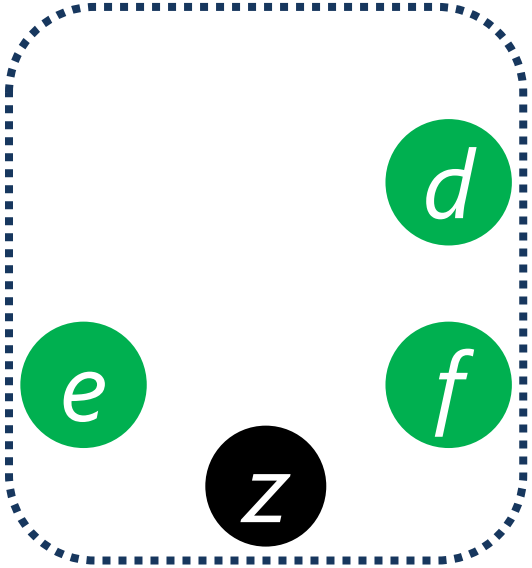
必要なもの

① **更新できる索引構造**
② **索引更新手法**
③ **影響解析クエリ手法**



更新できる索引構造

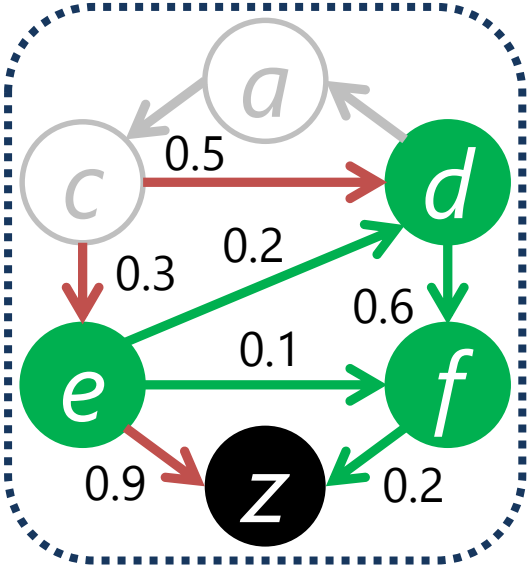
RISの素朴な適用



索引更新 **難**

情報過少
拡散経路が分からない

完全な情報

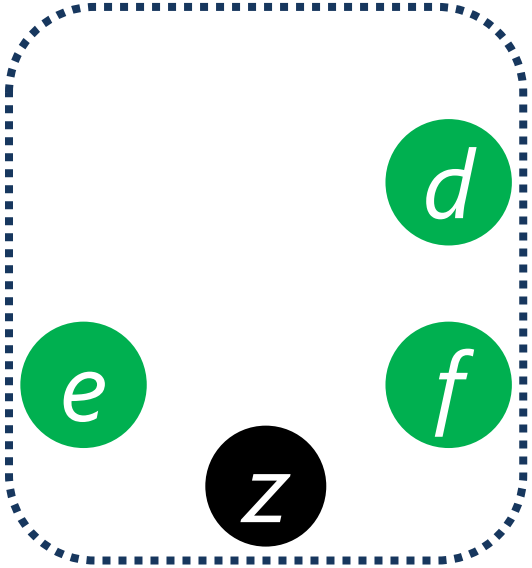


消費空間 **多**

300GB

更新できる索引構造

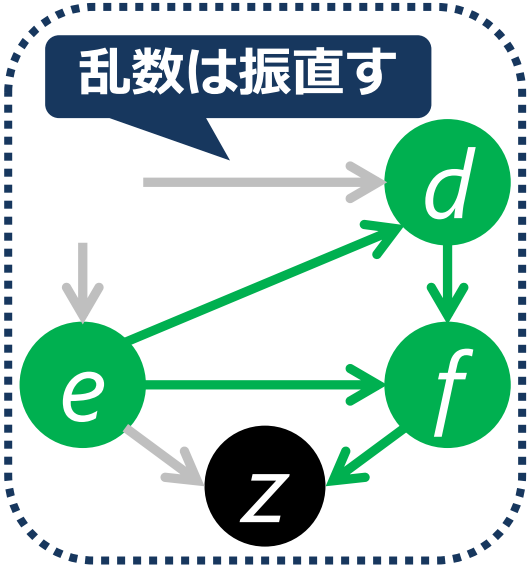
RISの素朴な適用



索引更新 **難**

情報過少
拡散経路が分からない

提案手法

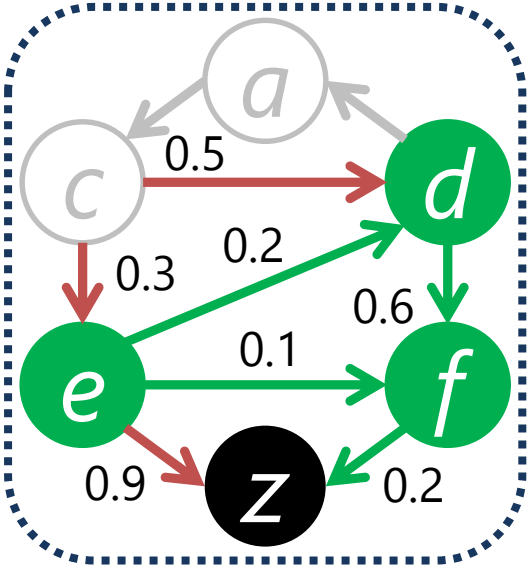


索引更新 **易**

消費空間 **少**

30GB

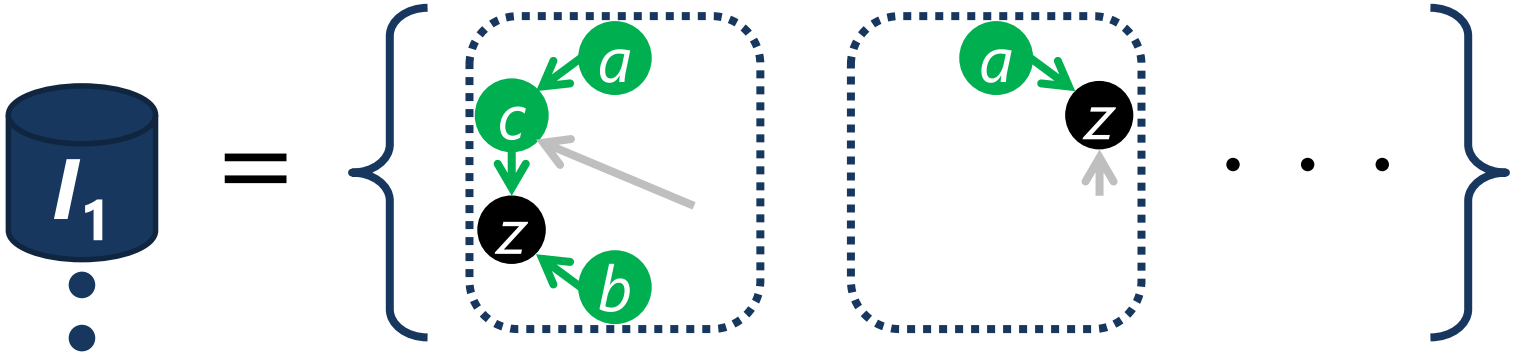
完全な情報



消費空間 **多**

300GB

索引更新手法の概要



辺追加・辺削除・辺確率変更・頂点追加・頂点削除

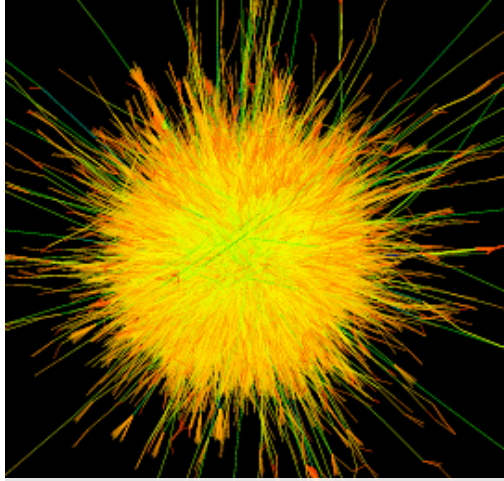
z に影響 (到達) する頂点集合を更新

辺削除のみ説明します

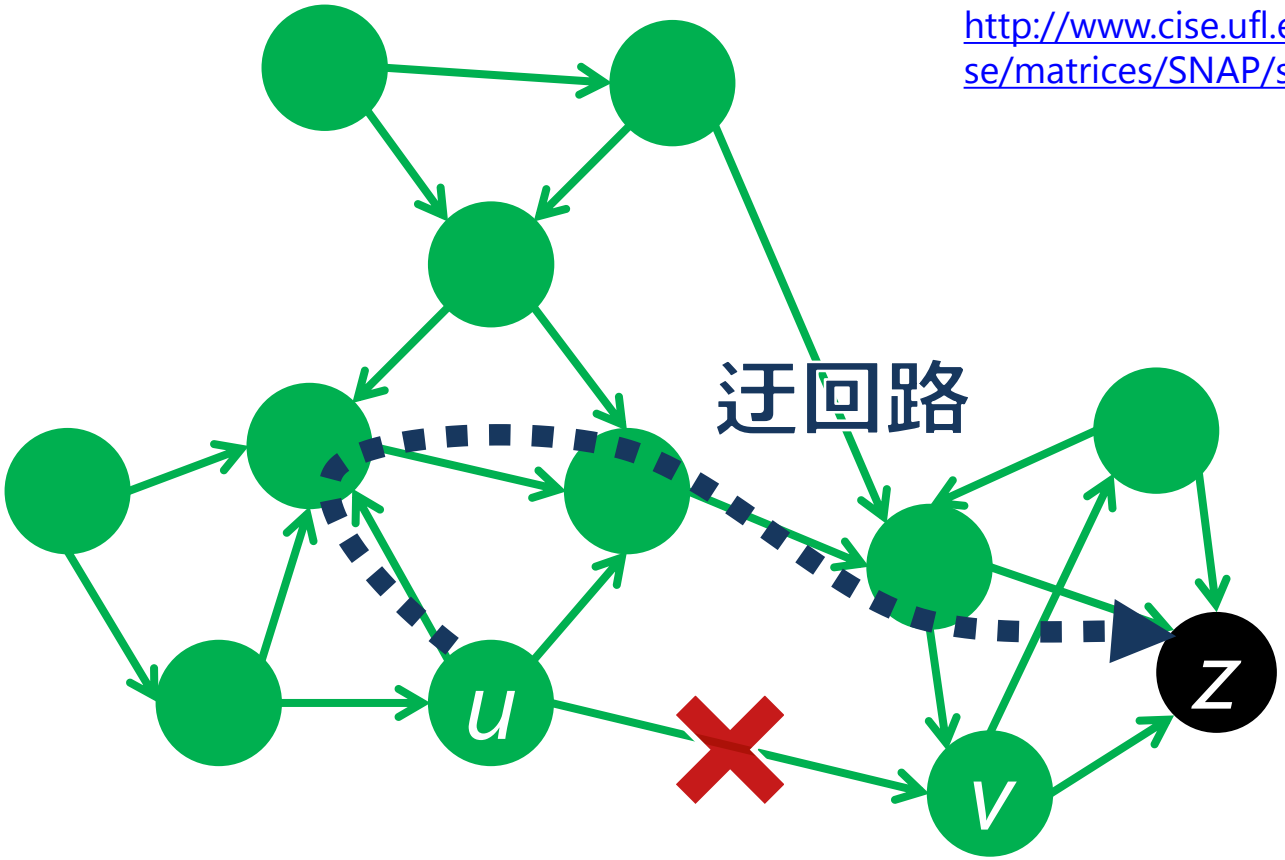
簡単のため，辺確率＝1 & 単一スケッチ

辺削除の例1

Q.「z に到達可能な頂点」が**減る**？



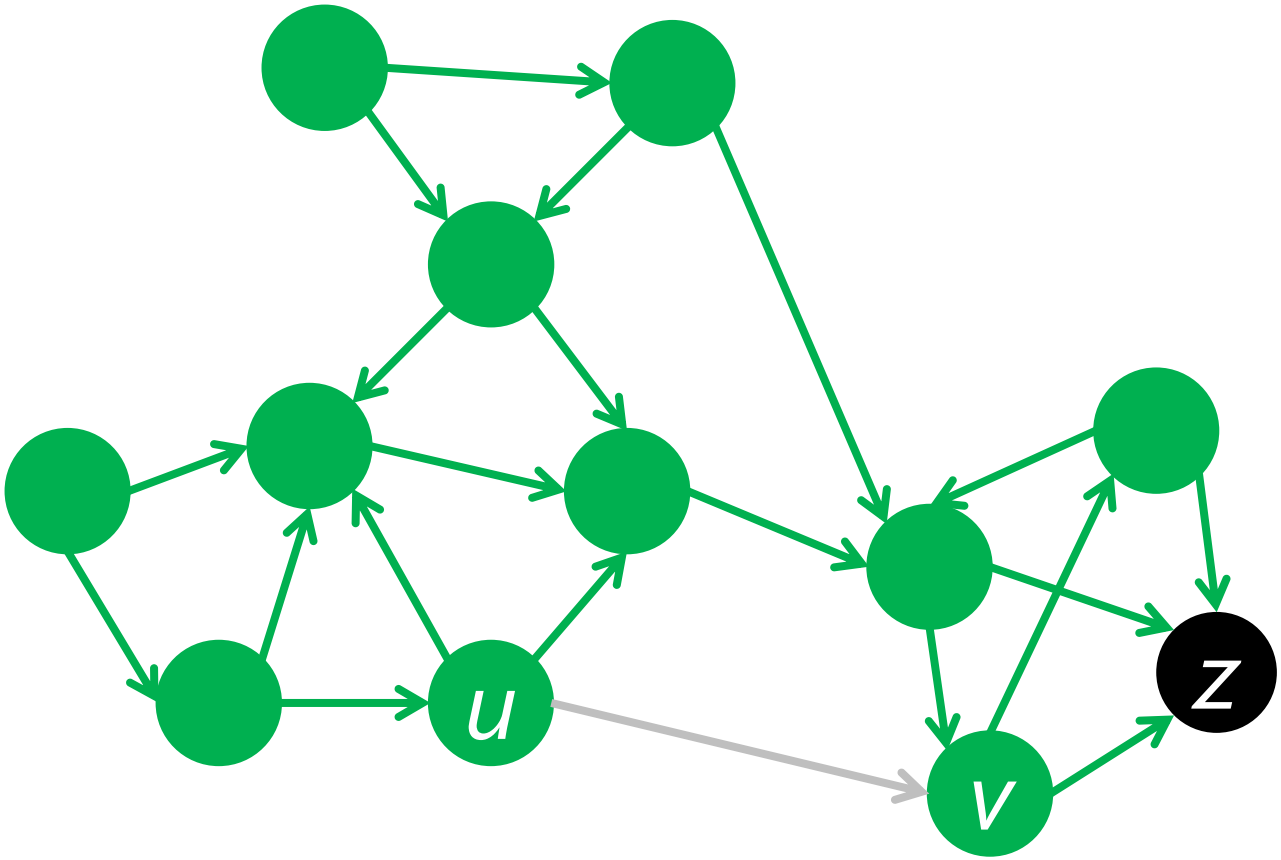
http://www.cise.ufl.edu/research/sparse/matrices/SNAP/soc-Epinions1.html



辺削除の例1

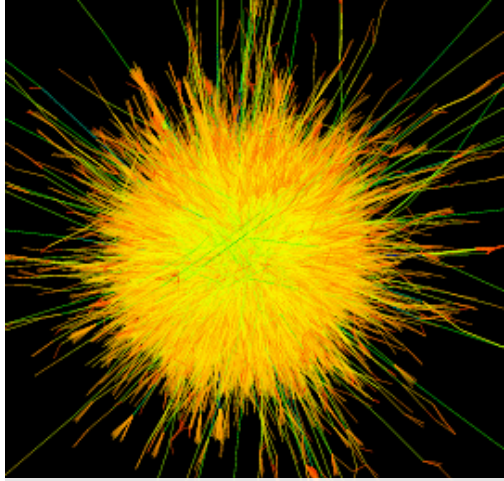
Q.「z に到達可能な頂点」が**減る**？

A. 減らない

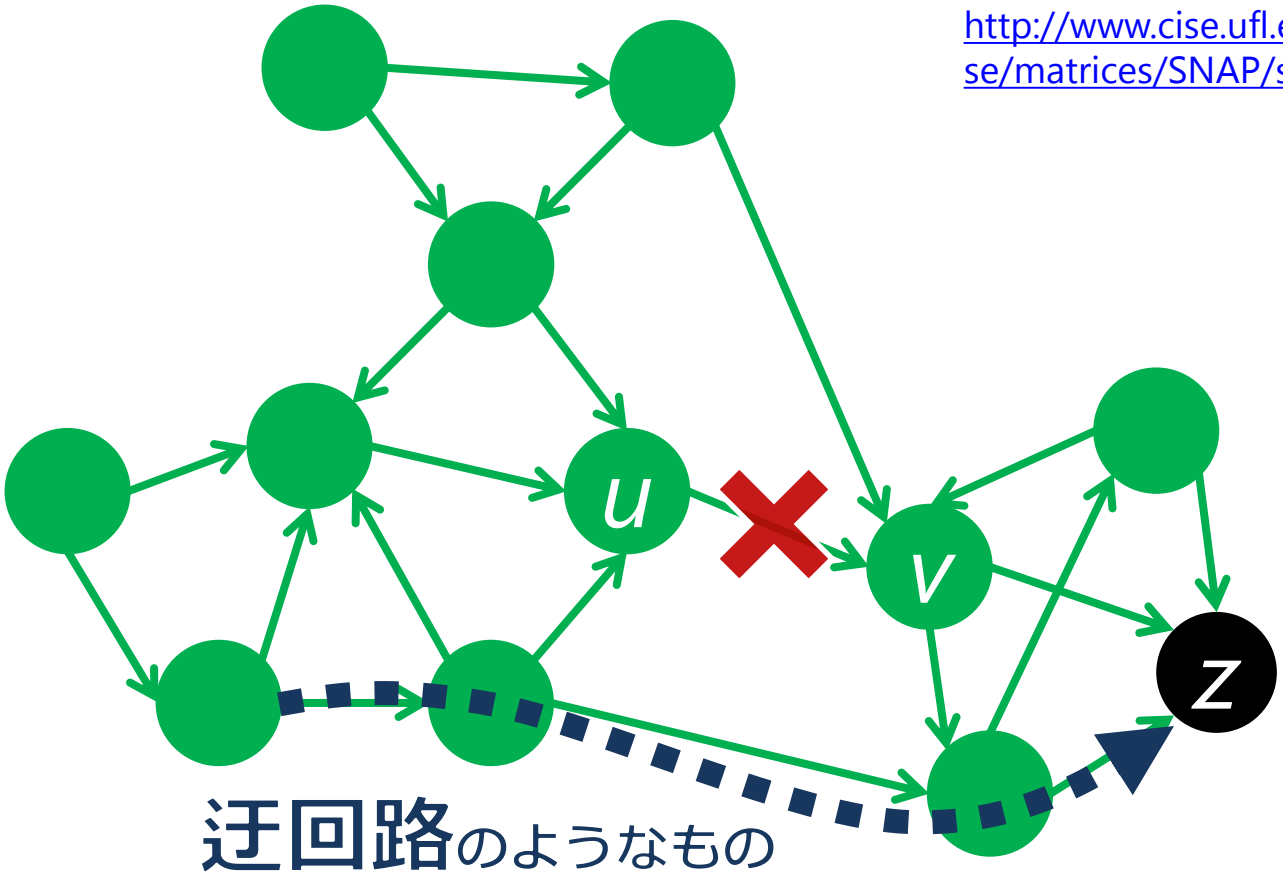


辺削除の例2

Q.「z に到達可能な頂点」が**減る**？



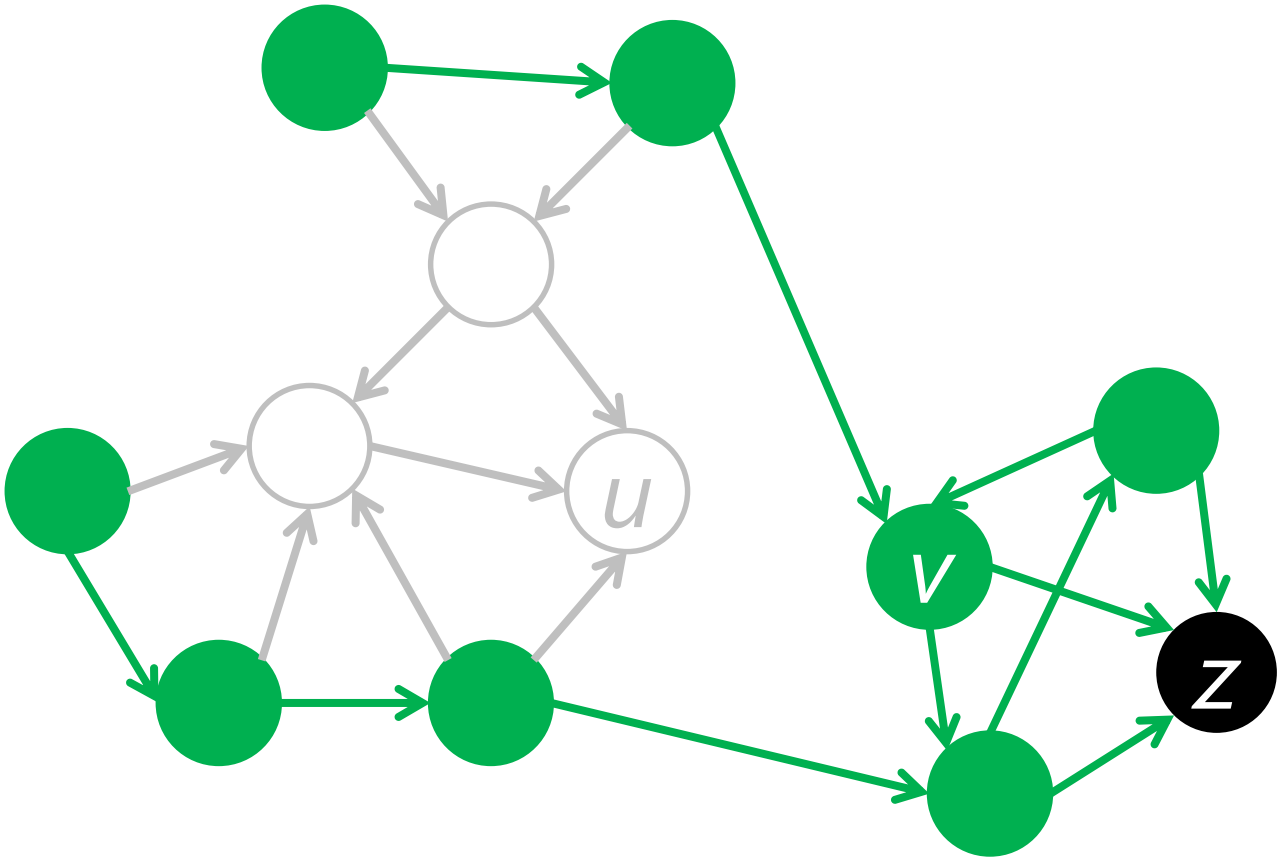
http://www.cise.ufl.edu/research/sparse/matrices/SNAP/soc-Epinions1.html



辺削除の例2

Q.「z に到達可能な頂点」が**減る**？

A. 少し減る

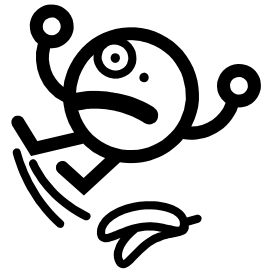


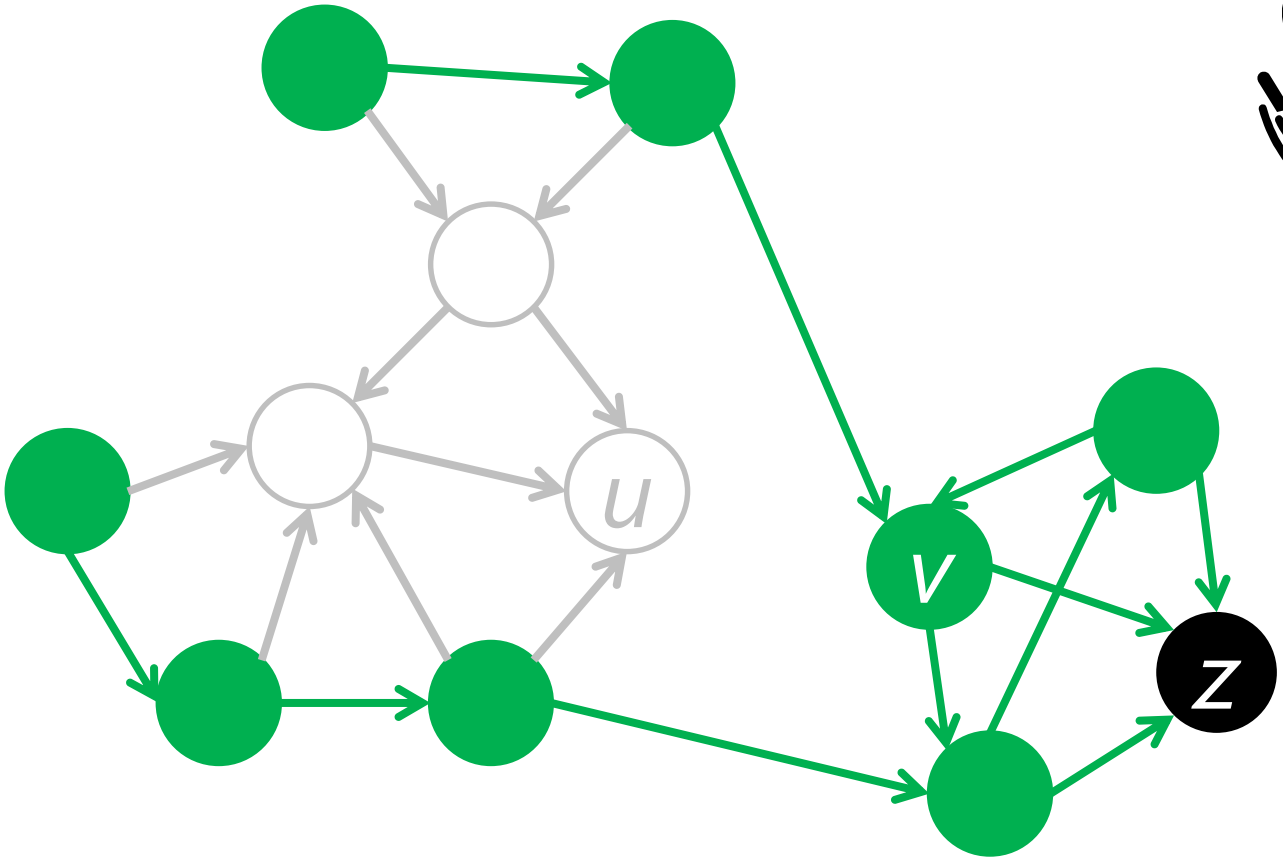
辺削除の素朴な更新方法

●z から逆幅優先探索で良いのでは？

○を消すため全●を見る

＼遅い／





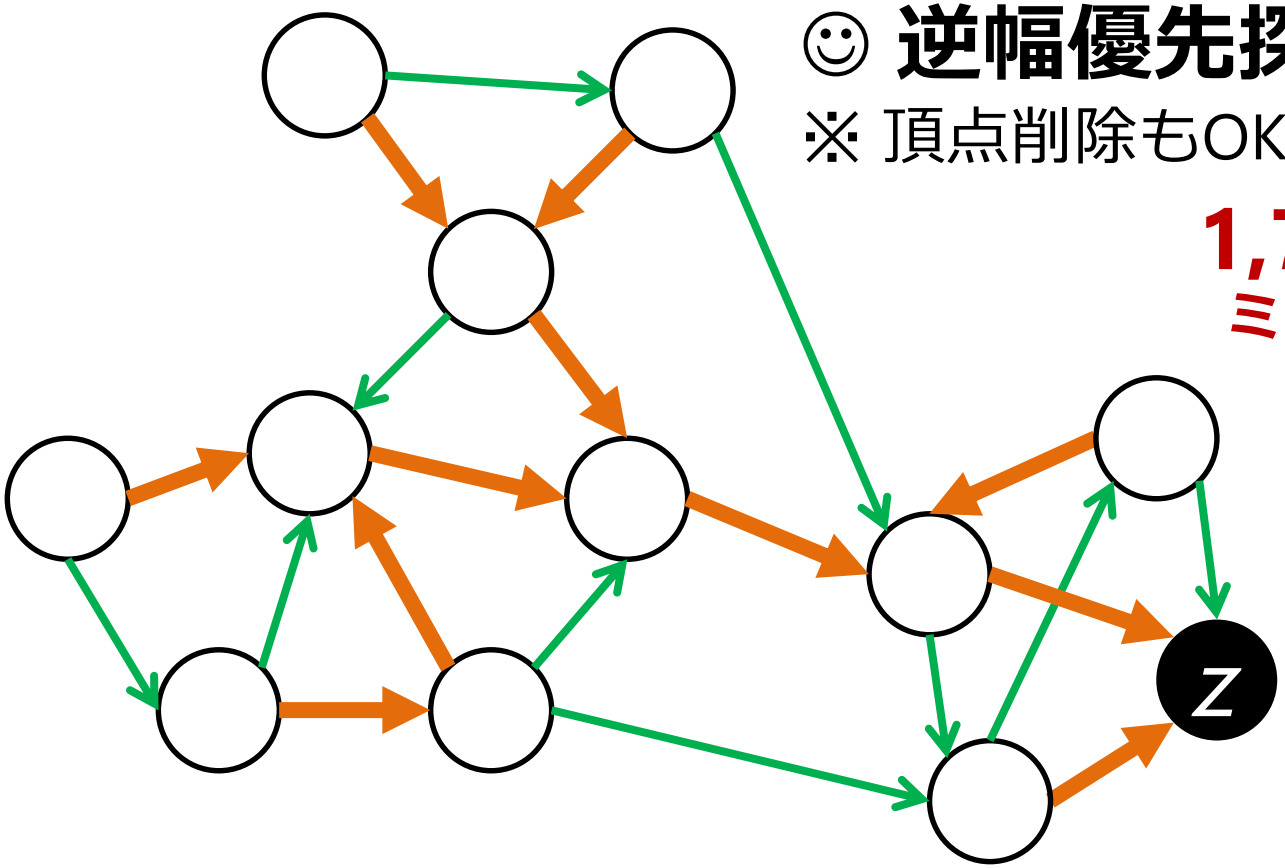
辺削除の高速な反映：**到達可能木**の導入

z を根とするスケッチの部分有向木

☺ **迂回路の存在判定**

☺ **逆幅優先探索の範囲抑制**

※ 頂点削除もOK

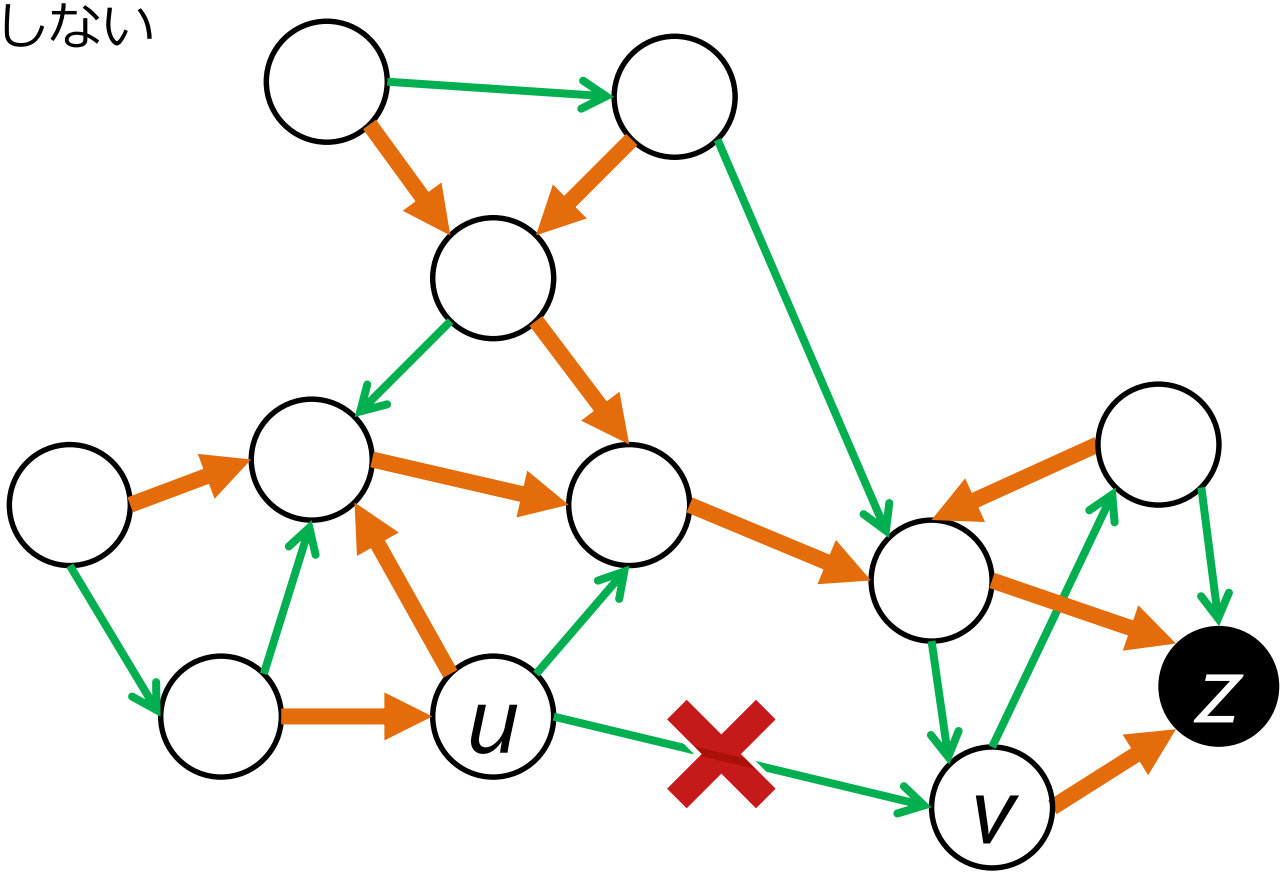


1,700 ミリ秒 ・・▶ **2** ミリ秒

辺削除の高速な反映：迂回路の存在判定

$uv \notin$ **到達可能木** ⇒ⓤから❷へ迂回路が有る

※逆は成立しない



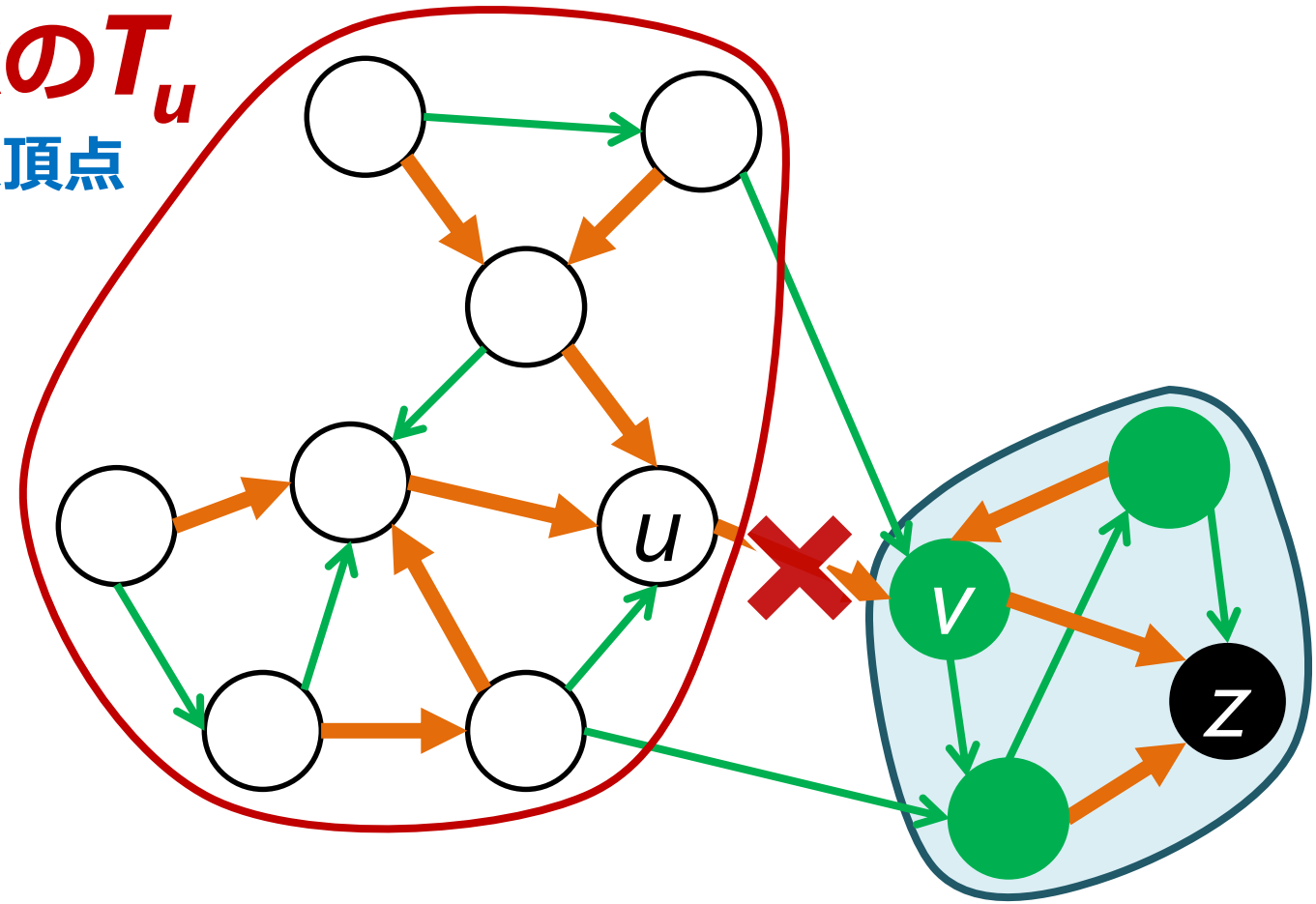
↗ の削除：何もしなくてOK

辺削除の高速な反映：探索範囲の抑制

ⓤを根とする部分木T_uだけ調査 & 木を更新

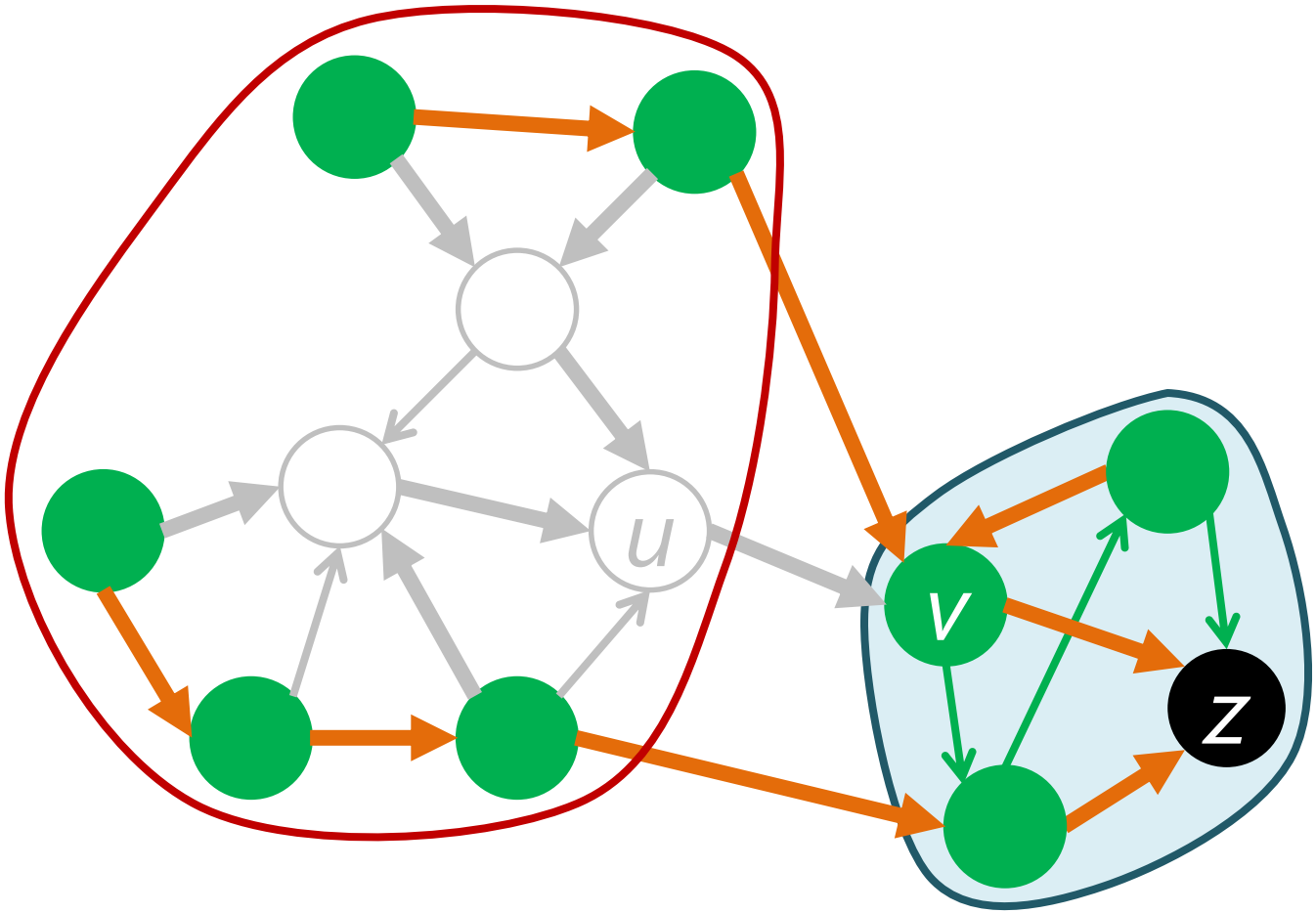
疑惑のT_u

高々**数頂点**



辺削除の高速な反映：探索範囲の抑制

ⓤを根とする部分木T_uだけ調査 & 木を更新



影響解析のクエリアルゴリズム

RIS [Borgs-Brautbar-Chayes-Lucier. *SODA'14*] をベースに

索引再構築不要を活かし効率化

- ハッシュテーブルによる動的管理
- Lazy greedyの適用 [Minoux. *Optimization Techniques'78*]

精度保証 索引サイズ $= \Theta(\epsilon^{-3}(|V| + |E|)\log|V|)$

- 影響力推定の精度　$\sigma(S) \pm \epsilon|V|$　*w.h.p.*　(定理 5.9)
- 影響最大化の近似比　$1 - \mathrm{e}^{-1} - \epsilon$　*w.h.p.*　(定理 5.10)
- 索引更新手法の非退化性 (定理5.8) ⇝ **再構築の必要無**

実験

索引構築
索引更新
影響力推定
影響最大化
} の効率

- データ：Koblenz Network Collection http://konect.uni-koblenz.de/
  辺の作成時刻付き
- 計算機：Intel Xeon E5-2690 2.90GHz CPU + 256GB RAM
- コンパイラ：g++v4.6.3 (-O2)
- 索引サイズ $= 32(|V| + |E|)\log|V|$

索引構築

実験設定		索引構築		完全な情報
ネットワーク	p	時間	サイズ	サイズ
Epinions 13万点 84万辺	①	89 s	1 GB	6 GB
	②	62 s	1 GB	7 GB
YouTube 322万点 1,875万辺	①	5,000 s	45 GB	250 GB
	②	1,986 s	4 GB	180 GB
Flickr 230万点 3,314万辺	①	5,468 s	31 GB	≈ 282 GB
	②	4,254 s	12 GB	≈ 292 GB

- 数時間だが一度きり

① 辺*uv*の確率 = 0.1, 0.01, 0.001から無作為に選択
② 辺*uv*の確率 = 入次数$(v)^{-1}$

グラフ変化による索引更新

実験設定		単一辺操作			単一頂点操作	
ネットワーク	p	追加	削除	確率変更	追加	削除
Epinions 13万点 84万辺	①	4.1 ms	1.0 ms	5.8 ms	0.8 ms	14.8 ms
	②	1.0 ms	1.8 ms	1.7 ms	0.7 ms	8.3 ms
YouTube 322万点 1,875万辺	①	31.8 ms	0.3 ms	236.2 ms	0.0 ms	92.2 ms
	②	0.1 ms	0.0 ms	1.5 ms	0.7 ms	5.7 ms
Flickr 230万点 3,314万辺	①	89.6 ms	2.4 ms	125.2 ms	0.0 ms	**459.0 ms**
	②	0.2 ms	0.1 ms	4.8 ms	2.1 ms	53.8 ms

- (更新時間) ≪ (構築時間)
- 頂点削除が最遅 ∵多量の辺削除を伴う
  但し，頻度は少ないと思われる

① 辺uvの確率 = 0.1, 0.01, 0.001から無作為に選択
② 辺uvの確率 = 入次数$(v)^{-1}$

単一頂点の影響力推定の時間

実験設定		本研究		静的手法	
ネットワーク	p	索引構築	クエリ	MC [Kempe+'03]	RIS [Borgs+'14]
Epinions 13万点 84万辺	①	89 s	**0.97 μs**	6 s	9 s
	②	62 s	**0.96 μs**	0.01 s	9 s
YouTube 322万点 1,875万辺	①	5,000 s	**1.79 μs**	> 100 s	519 s
	②	1,986 s	**1.68 μs**	0.02 s	447 s
Flickr 230万点 3,314万辺	①	5,468 s	**1.83 μs**	> 100 s	350 s
	②	4,254 s	**1.74 μs**	0.05 s	473 s

- 100万点／秒の追跡可能
- 実は，表引きしてるだけ

① 辺uvの確率 = 0.1, 0.01, 0.001から無作為に選択
② 辺uvの確率 = 入次数$(v)^{-1}$

影響最大化の時間 (シードサイズ $k = 100$)

実験設定		本研究	静的手法			
ネットワーク	p	クエリ	RIS [Borgs+'14]	IMM [Tang+'15]	PMC [Ohsaka+'14]	IRIE [Jung+'12]
Epinions 13万点 84万辺	①	**0.5 s**	10 s	39 s	11 s	13 s
	②	0.4 s	12 s	**0.3 s**	21 s	13 s
YouTube 322万点 1,875万辺	①	**23 s**	508 s	メモリ不足	284 s	250 s
	②	**1 s**	535 s	8 s	922 s	239 s
Flickr 230万点 3,314万辺	①	**16 s**	361 s	メモリ不足	173 s	497 s
	②	**3 s**	617 s	6 s	932 s	457 s

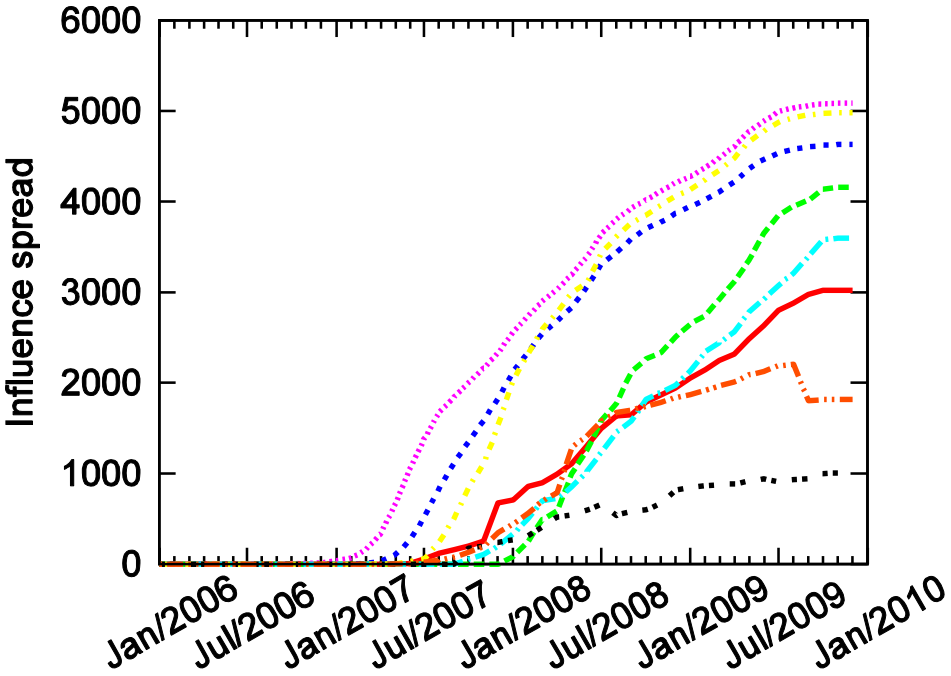
▶ スケッチが既にある効果 ← **本研究**と同精度設定

① 辺uvの確率 = 0.1, 0.01, 0.001から無作為に選択
② 辺uvの確率 = 入次数$(v)^{-1}$

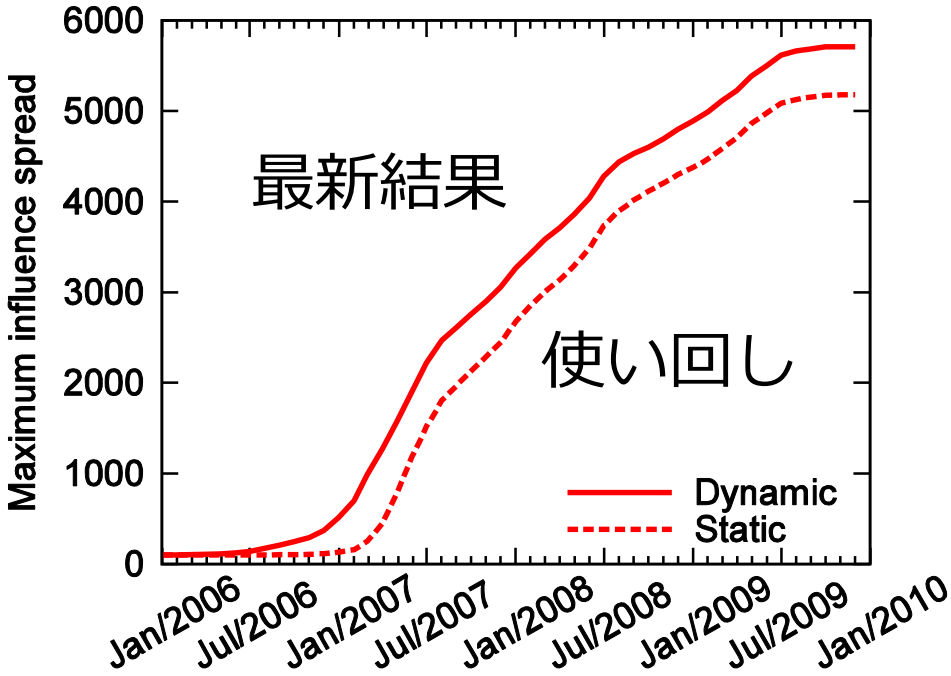
実際にできること

Flixster (映画レビューサイト)

頂点の影響力の遷移



100頂点集合の最大影響力の遷移

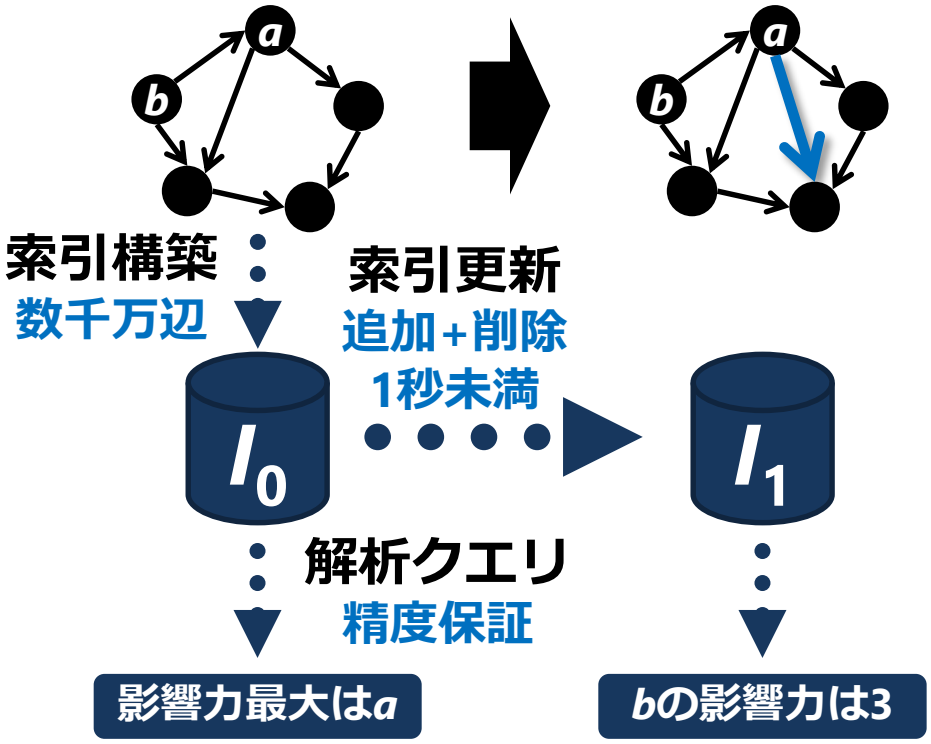


データセット http://www.cs.ubc.ca/~jamalim/datasets/

まとめ

完全動的索引手法を提案

- 動的グラフ上の
  影響解析クエリを実現



研究を通じ感じた今後

- 索引の省スペース化
  圧縮できるか？
- 影響最大化クエリの高速化
  動的な設定でMaximum Coverage
  劣線形時間でできるか？