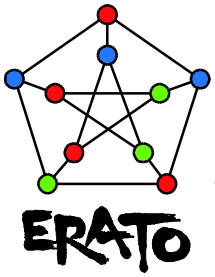
完全動的索引によるグラフ上の影響力推定・影響最大化クエリ

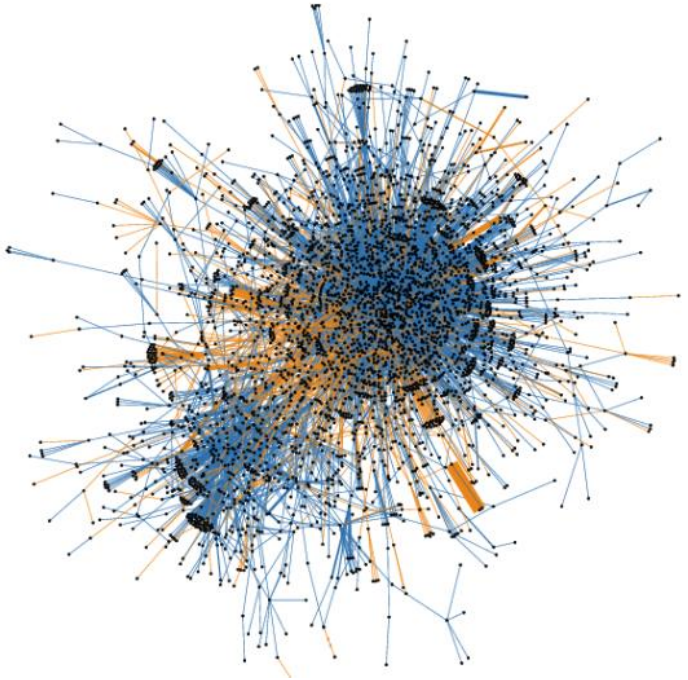
大坂 直人（東京大学）
秋葉 拓哉（NII）
吉田 悠一（NII & PFI）
河原林 健一（NII）



河原林巨大グラフプロジェクト
Kawarabayashi Large Graph Project

ネットワーク上の拡散

- ニュース・世論・噂 [Valente. 1995]
- 口コミ [Brown, Reinegen. 1987]
- 感染症 [Bailey. 1975]
- 水質汚染 [Ostfeld, Uber, Salomons. 2006]
- Twitterにおけるリツイート



http://cnets.indiana.edu/wp-content/uploads/tcot.png

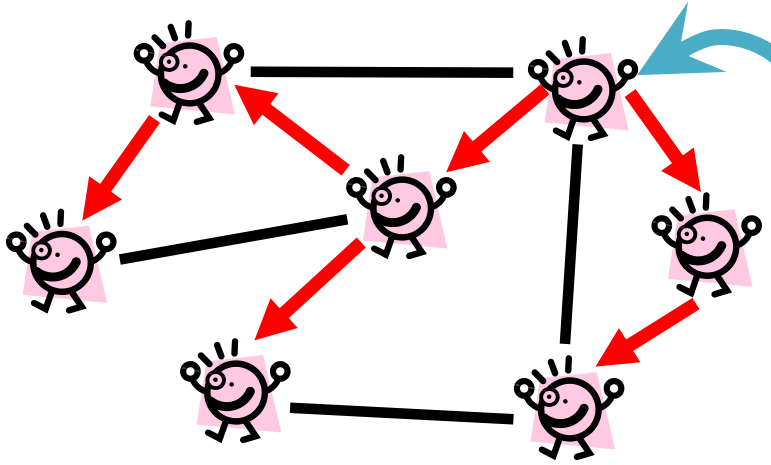
近年の注目

ソーシャルネットワーク上の情報拡散

計算機科学・統計物理学等の分野において

バイラルマーケティング

[Domingos, Richardson. KDD'01], [Richardson, Domingos. KDD'02]



無料／割引商品

目標: 費用を抑えて
利益を最大化

Q. 各頂点の影響力は？ → **影響力推定**

Q. 最も影響力の高い頂点集合は？

→ **影響最大化**

離散最適化問題

[Kempe, Kleinberg, Tardos. KDD'03]

これまでの研究－いかに速く解くか？

影響最大化 (近似)：$\mathcal{O}$(**頂点数** × **辺数** × #**Sim.**)時間

- ’06年~’14年で進歩
- 数千万辺を数百秒

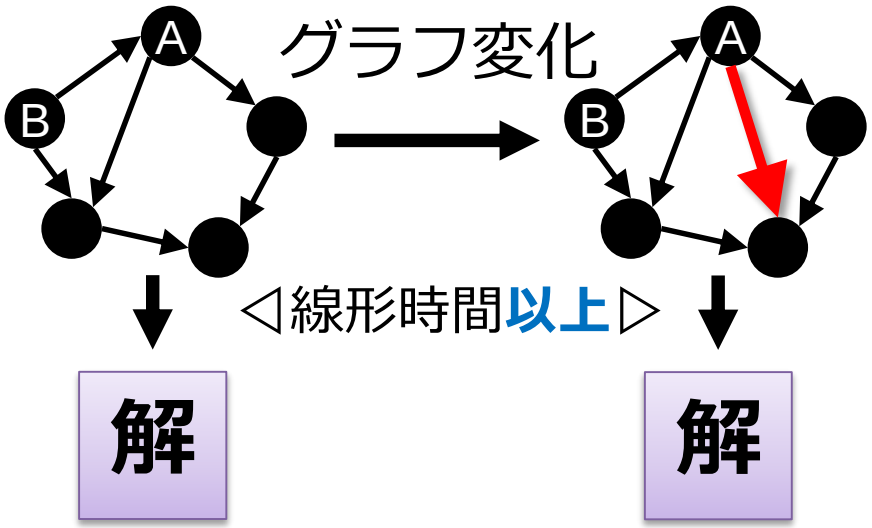
[Ohsaka, Akiba, Yoshida, Kawarabayashi. AAAI’14]
[大坂, 秋葉, 吉田, 河原林. DEIM’14]

影響力推定 (近似)：$\mathcal{O}$(**辺数** × #**Sim.**)時間

現実のグラフは**動的**・**大規模**

Twitter　6,000Tweets/秒　3億ユーザ

https://about.twitter.com/company/

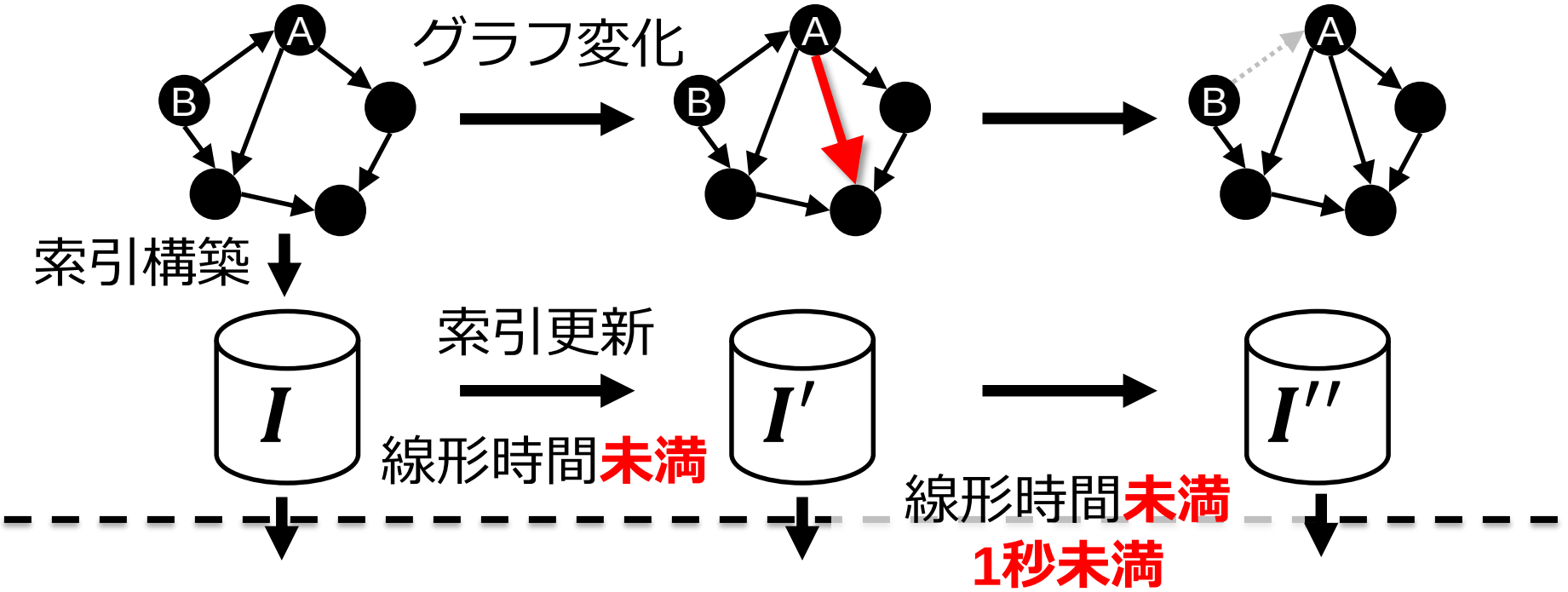


グラフサイズの**線形時間未満**へ

本研究の貢献

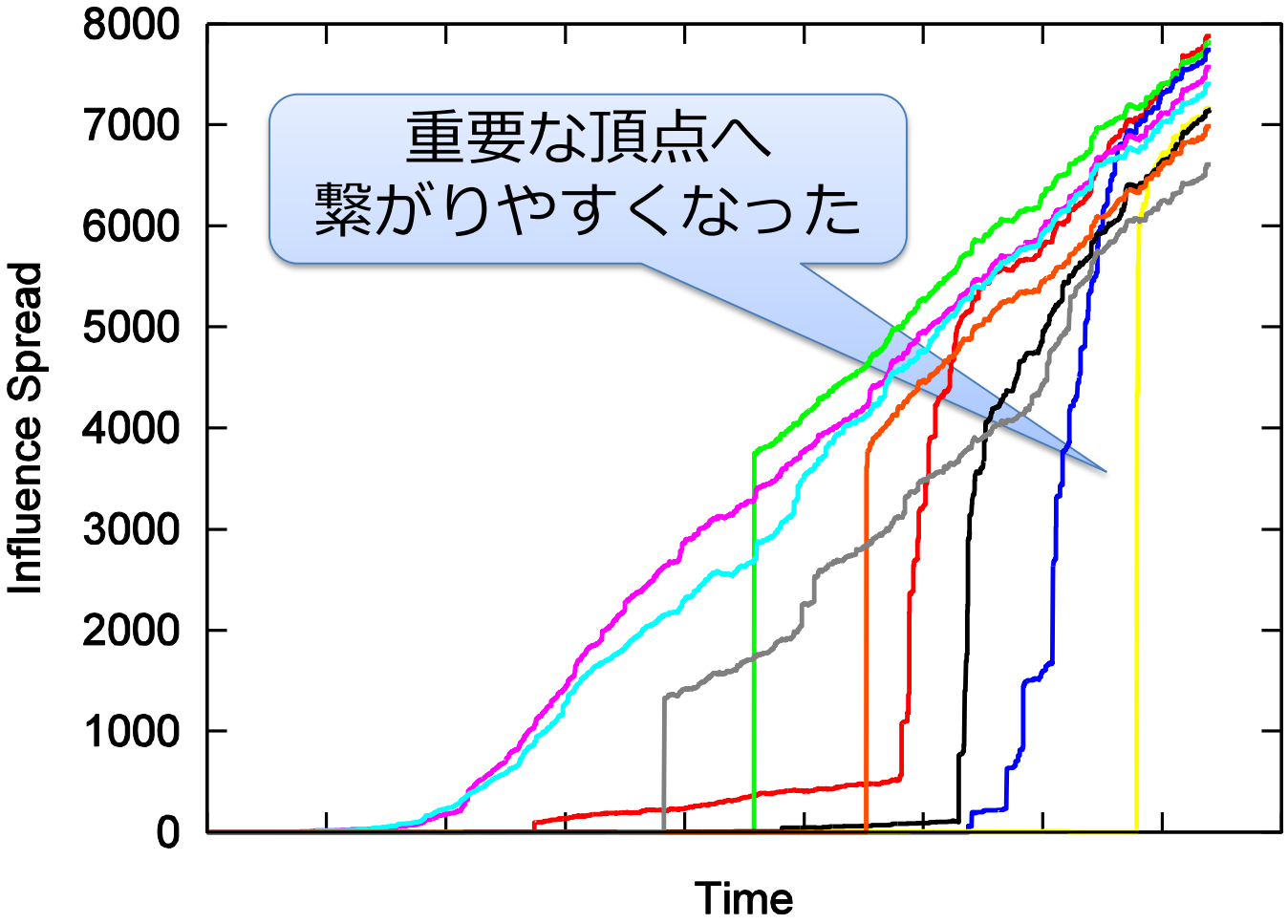
動的グラフ上の影響力推定・影響最大化クエリの完全動的索引データ構造の提案

辺追加・辺削除・辺確率変更・頂点追加・頂点削除



できること

成長するネットワーク(Epinions.com)上の
頂点の影響力の軌跡

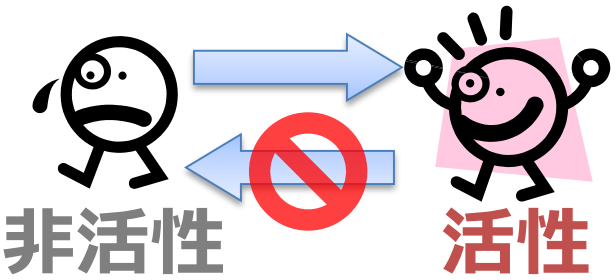


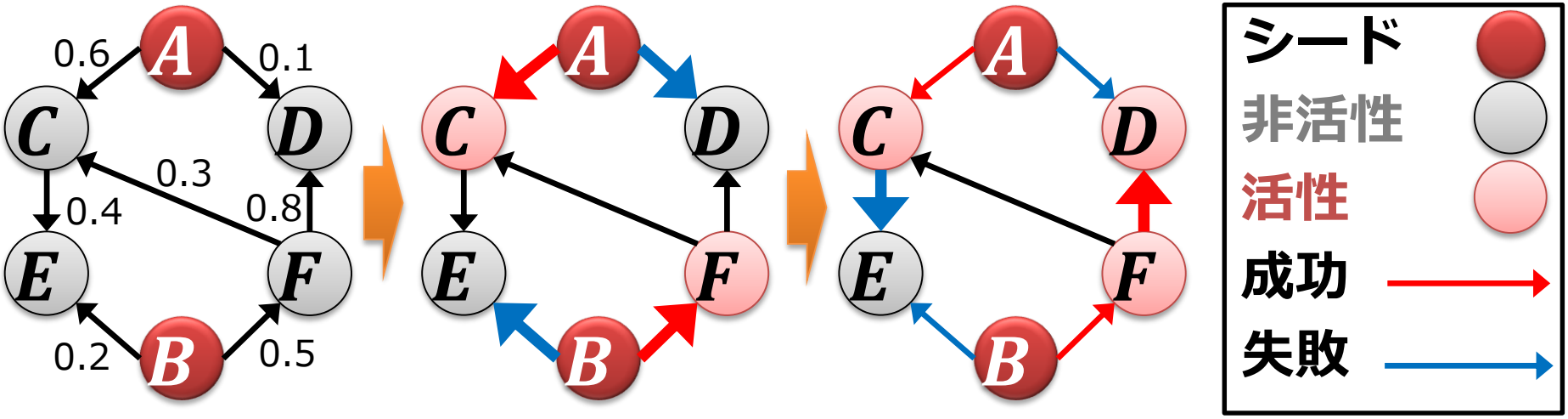
独立カスケードモデル

[Goldenberg, Libai, Muller. Market. Lett.'01]

- **入力**
  - グラフ $G = (V, E)$
  - 辺確率 p_e $(e \in E)$
  - シード $S \subseteq V$

- **頂点の状態**





影響拡散 $\sigma(S)$：
Sをシードとして**活性化**する頂点数の**期待値**

取り組む問題と既存結果

問題1 (**影響力推定**)
入力：$S \subseteq V$
出力： $\sigma(S)$

厳密計算は **#P-hard**
[Chen, Wang, Wang. KDD'10]

Monte-Carloシミュレーションにより近似可能

問題2 (**影響最大化**)
[Kempe, Kleinberg, Tardos. KDD'03]
入力：$k \in \mathbb{N}$
出力：$\underset{S:|S|=k}{\operatorname{argmax}}\ \sigma(S)$

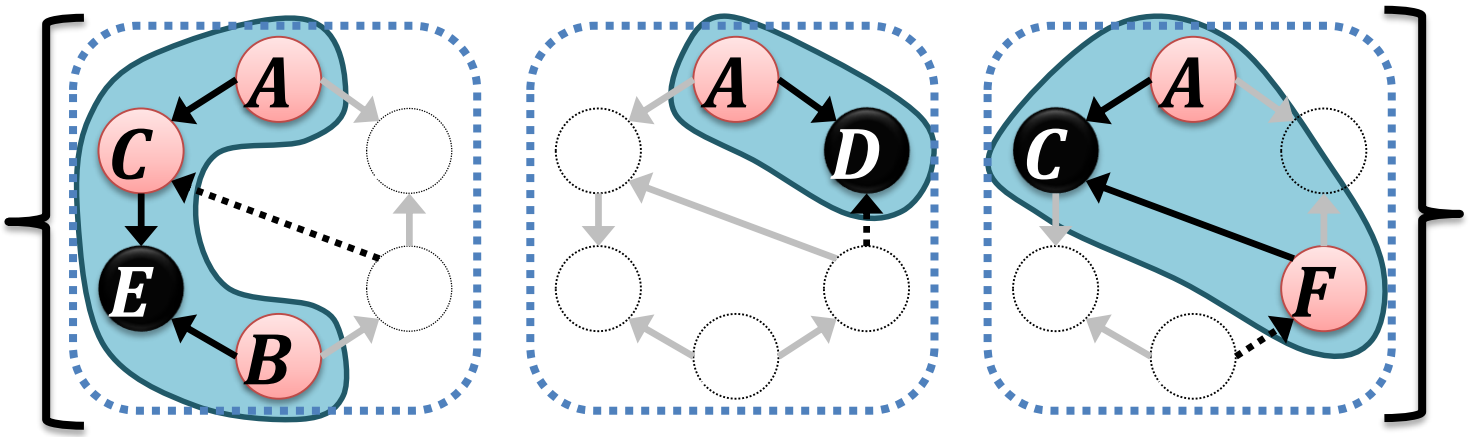
厳密計算は **NP-hard**
[Kempe, Kleinberg, Tardos. KDD'03]

貪欲アルゴリズムが
$\left(1-\frac{1}{e}\right) \approx 63\%$ 近似
$\sigma(\cdot)$の**劣モジュラ性**より
[Nemhauser, Wolsey, Fisher. Math. Program.'78]

いかに$\sigma(\cdot)$を**高速**・**精確**に見積もるか？

スケッチ手法 [Borgs, Brautbar, Chayes, Lucier. SODA'14]

シードを**ランダム**に選び **逆シミュレーション**



A, B, CはEに影響　　AはDに影響　　A, FはCに影響

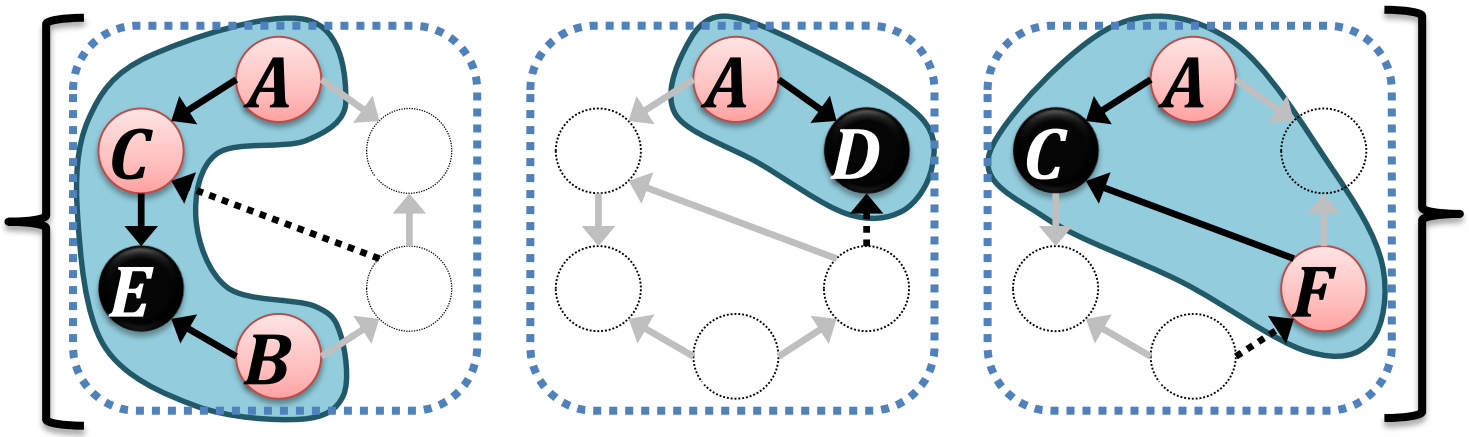
1. 頂点zを一様ランダムに選択
2. 逆グラフ上の拡散過程をzからシミュレート
3. スケッチ = 訪れた頂点集合

探索した辺数 $\leq R = \Theta(\epsilon^{-3}(|V| + |E|)\log|V|)$

■予備知識

スケッチ手法 [Borgs, Brautbar, Chayes, Lucier. SODA'14]

シードを**ランダム**に選び **逆シミュレーション**



観察: 「スケッチに多く現れる頂点は影響力が高そう」

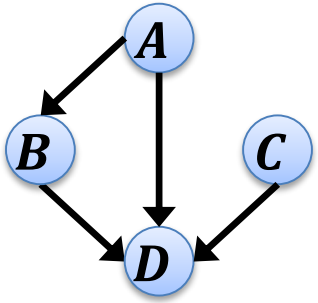
影響力推定・影響最大化クエリは

$\sigma(\{v\}) \propto \mathbf{E}[v$を含むスケッチ数$]$

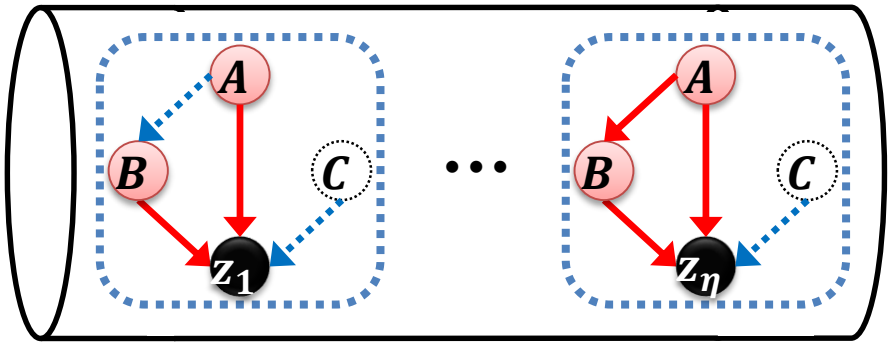
を元に近似計算可能

■提案手法

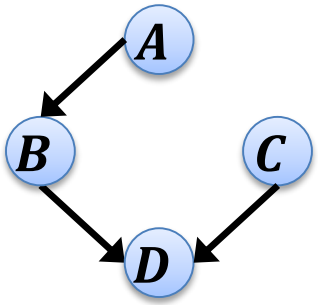
概観



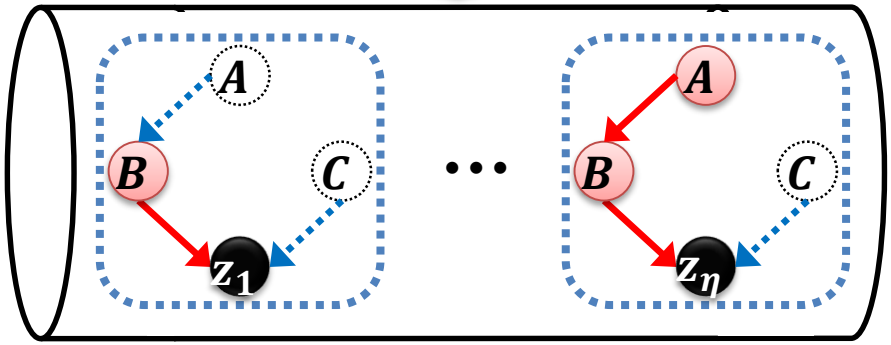
索引構築



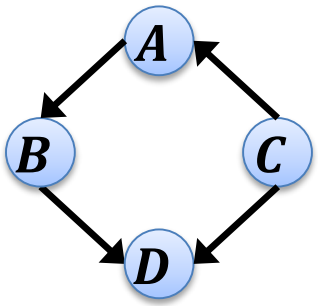
AD削除



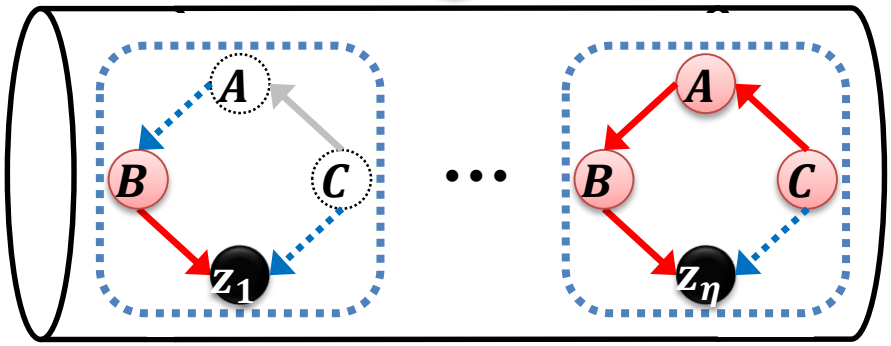
索引更新



CA追加



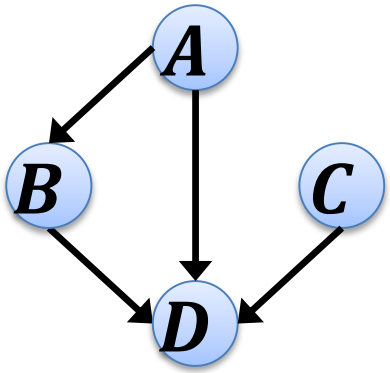
索引更新



■提案手法

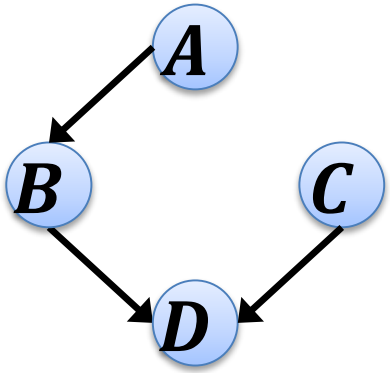
動的更新に スケッチ = {訪れた頂点} は不十分

元のグラフG_1

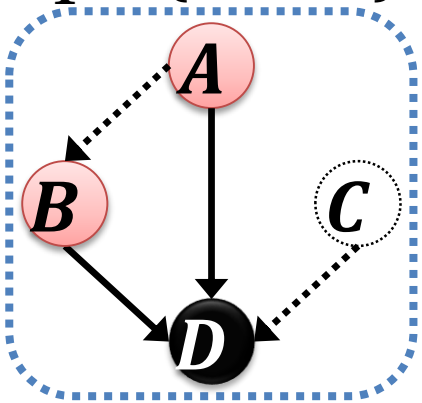


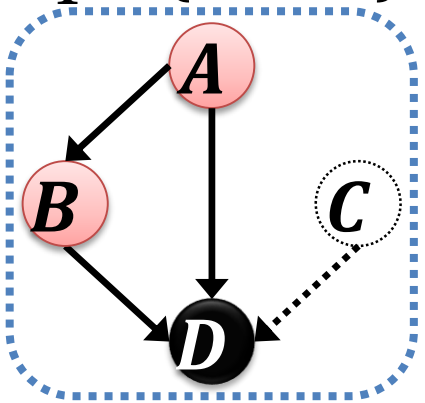
AD削除

元のグラフG_2

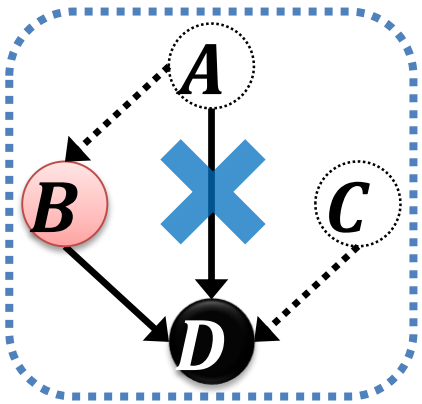


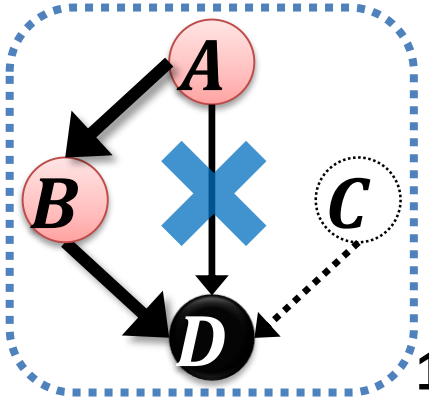
$A_1 = \{A, B, D\}$ ◀同じ▶ $B_1 = \{A, B, D\}$





$A_2 = \{B, D\}$ ◀違う▶ $B_2 = \{A, B, D\}$

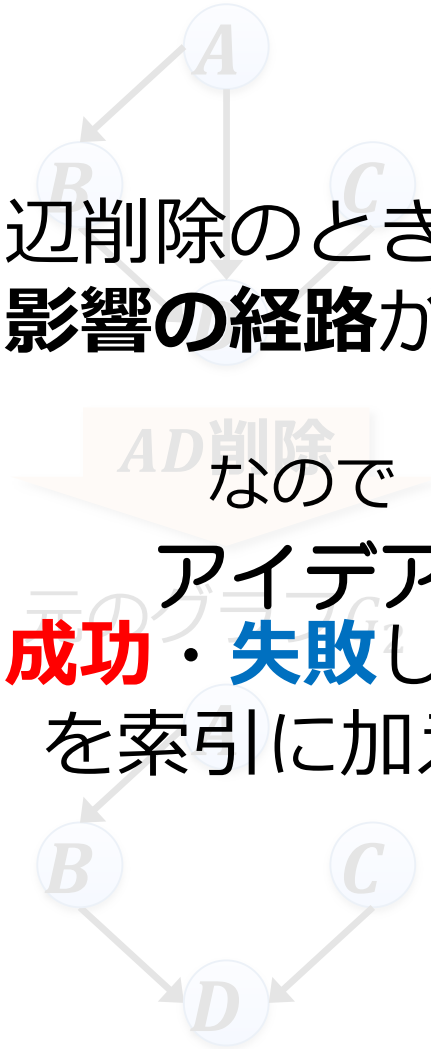




■提案手法

動的更新に スケッチ = {訪れた頂点} は不十分

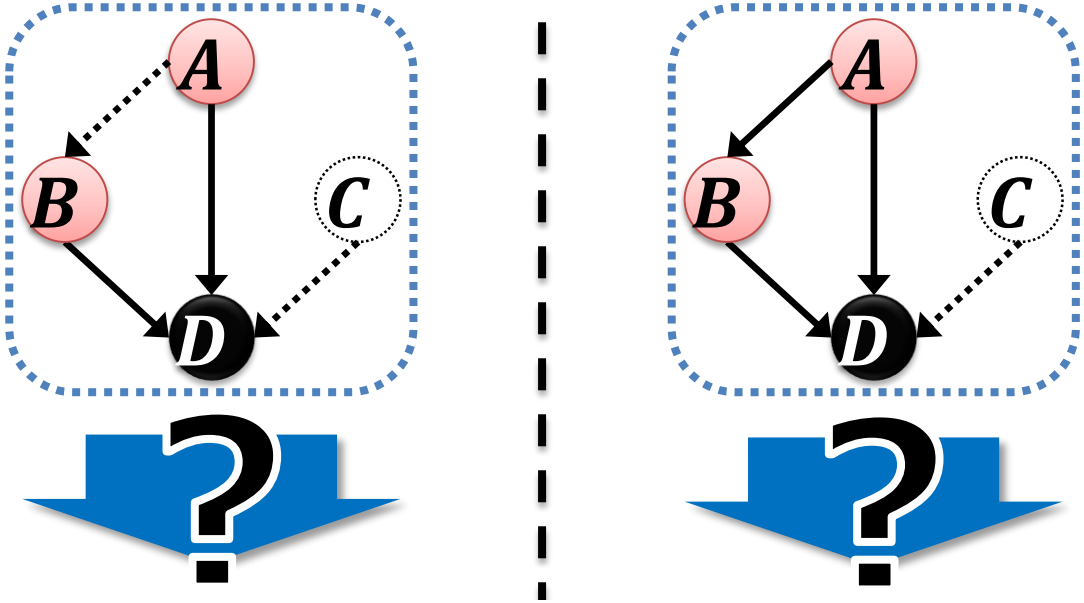
元のグラフG_1



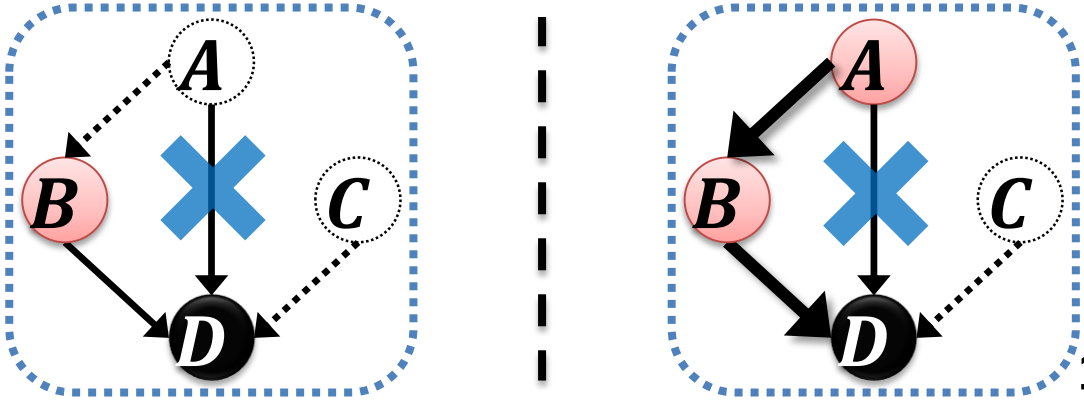
辺削除のときには
影響の経路が必要

なので
アイデア
成功・**失敗**した辺
を索引に加える

$A_1 = \{A, B, D\}$ ◀同じ▶ $B_1 = \{A, B, D\}$



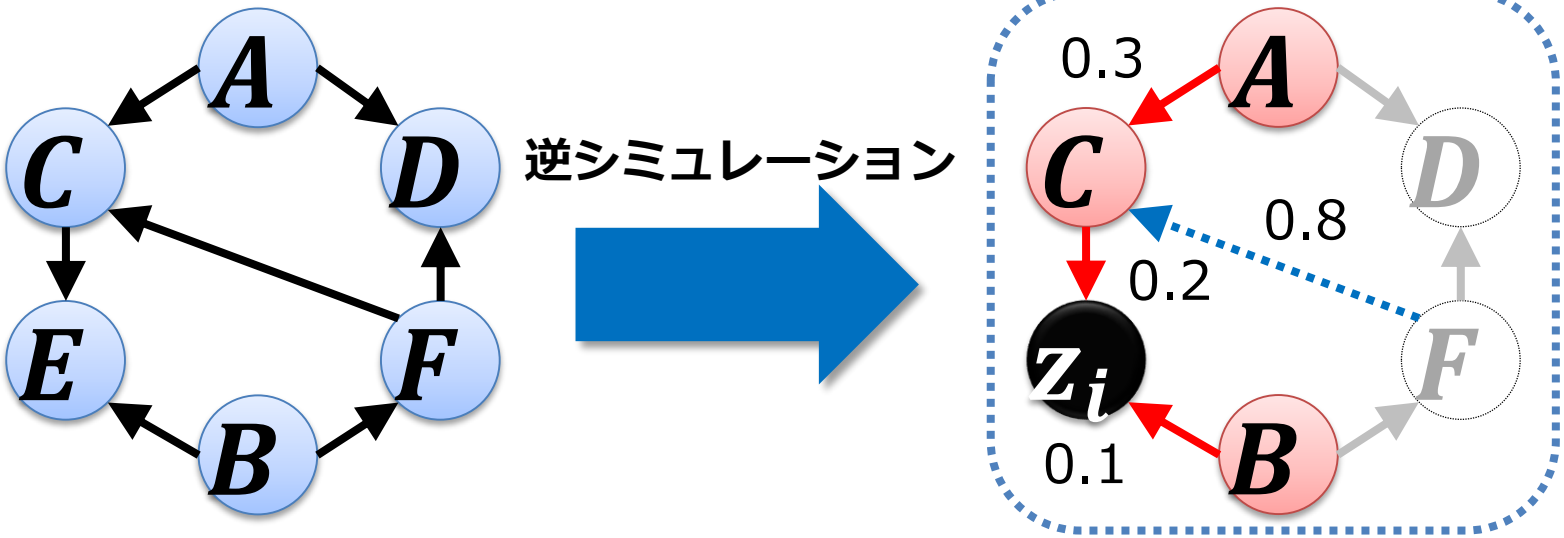
$A_2 = \{B, D\}$ ◀違う▶ $B_2 = \{A, B, D\}$



■提案手法

索引構造・索引構築

一つのスケッチ (z_i, x_i, H_i)



$I = \{(z_i, x_i, H_i)\}_i$

- z_i：**目標頂点**
- x_i：辺に振った乱数
- H_i：Gの部分グラフ

$E(H_i)$の辺の状態

- **成功**：$x_i(e) < p_e$な辺e
- **失敗**：$x_i(e) \geq p_e$な辺e
- **無意**：影響の経路に関わらない

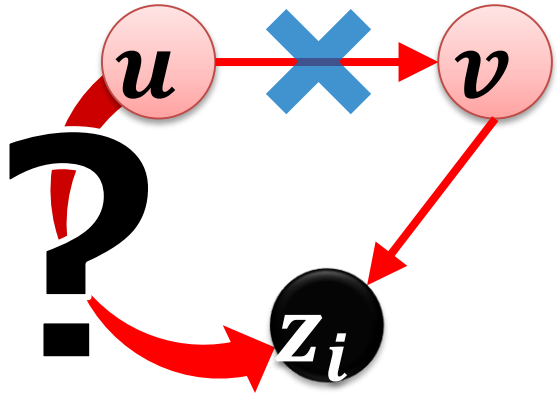
各スケッチの

成功辺でz_iに**到達可能**な$V(H_i)$をうまく更新

辺削除によるスケッチ更新

- クエリ：辺(u, v)を削除

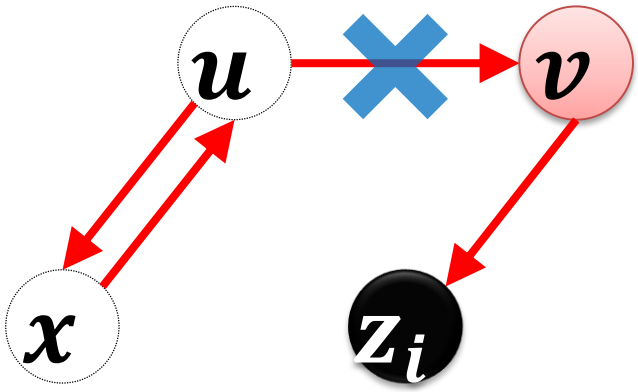
(uからz_iへの経路があったときに)
「(u, v)を通らないuからz_iへの**成功**経路があるか？」



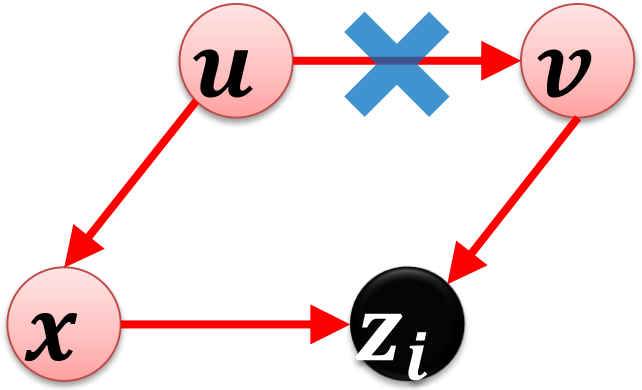
辺削除によるスケッチ更新

- クエリ：辺(u, v)を削除

(uからz_iへの経路があったときに)
「(u, v)を通らないuからz_iへの**成功**経路があるか？」



$u \to x \to \cdots \to z_i$ **NG**



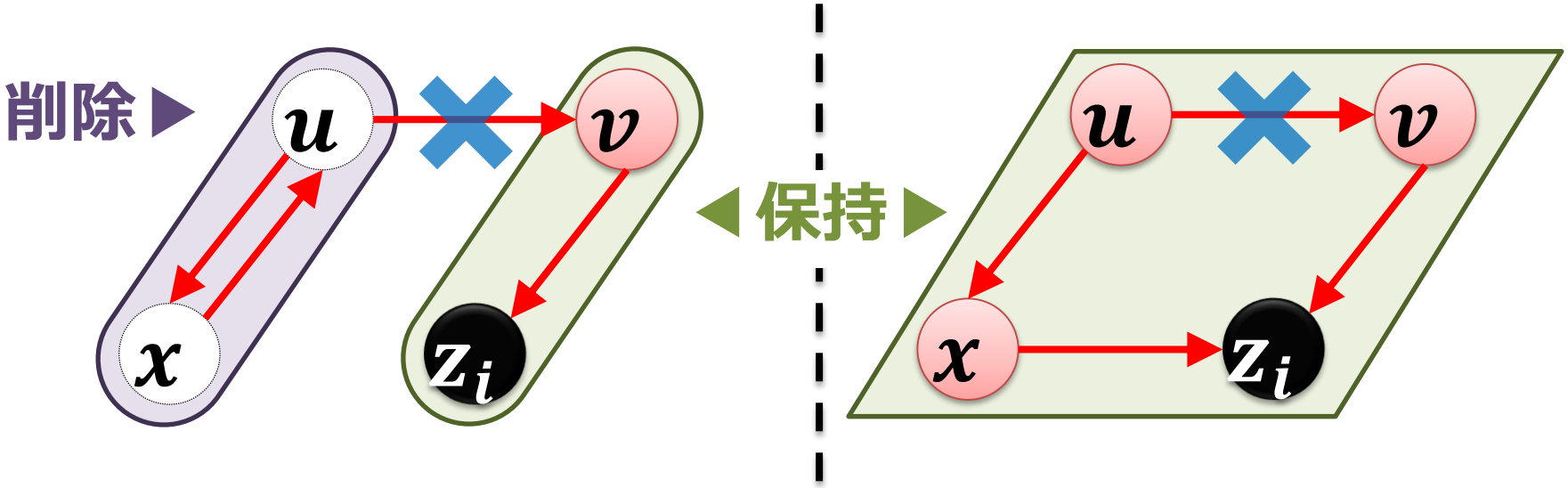
$u \to x \to z_i$ **OK**

uの近傍のみからの判定は難しい・・・

辺削除によるスケッチ更新

- クエリ：辺(u, v)を削除

(uからz_iへの経路があったときに)
「(u, v)を通らないuからz_iへの**成功**経路があるか？ 」

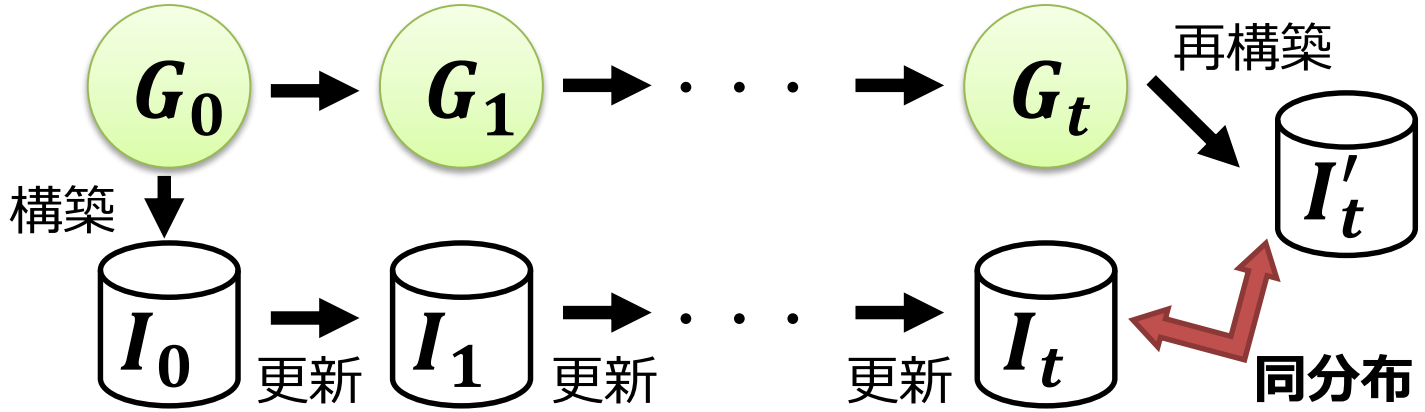


1. (u, v)を除きz_iから**成功**辺のみで**逆幅優先探索**
2. 未訪問の頂点を削除 ⇨ H_iが収縮

■提案手法

理論的解析

定理 11. 動的更新による索引の非退化性



以下が**任意の時点**で成立

(H_iの頂点数・辺数の総和)$= \Theta(\epsilon^{-3}(|V| + |E|)\log|V|)$

定理 3. 影響力推定 [Borgs, Brautbar, Chayes, Lucier. SODA'14]

高確率で $\sigma(S) \pm \epsilon n$ の範囲の推定値を返す

定理 4. 影響最大化 [Borgs, Brautbar, Chayes, Lucier. SODA'14]

高確率で近似比 $\left(1 - \frac{1}{e} - \epsilon\right)$ の解を返す

■実験

索引構築・更新時間 [s]

	Epinions $\|V\|$=114K, $\|E\|$=717K	Facebook $\|V\|$=64K, $\|E\|$=1.6M
索引構築	320	750
1頂点追加	**0.003**	**0.01**
1頂点削除	**6**	**50**
1辺追加	**0.4**	**0.5**
1辺削除	0.6	1.7
辺確率変更	0.6	0.9

- 追加：**1秒未満**　　索引構築の**100倍高速**
- 削除：**数十秒**　　高コストな逆幅優先探索が原因

データセットはKONECT: The Koblenz Network Collectionより
Environment: Intel Xeon X5670 (2.93GHz), 48GB, Language: C++

■実験

影響解析クエリの処理時間

Epinions $|V|$=114K, $|E|$=717K

影響力推定 (1頂点あたり)

提案手法	**0.024 ms**
スケッチ手法 [Borgs+14]	6.7 s
シミュレーション	3.6 s

影響最大化 (50頂点選択)

提案手法	**0.4 s**
スケッチ手法 [Borgs+14]	7.6 s
PMC [Ohsaka+14]	8.4 s
IRIE [Jung+12]	9.7 s

100,000頂点／秒の影響力の追従が可能

既存手法の10倍高速

データセットはKONECT: The Koblenz Network Collectionより
Environment: Intel Xeon X5670 (2.93GHz), 48GB, Language: C++

まとめ

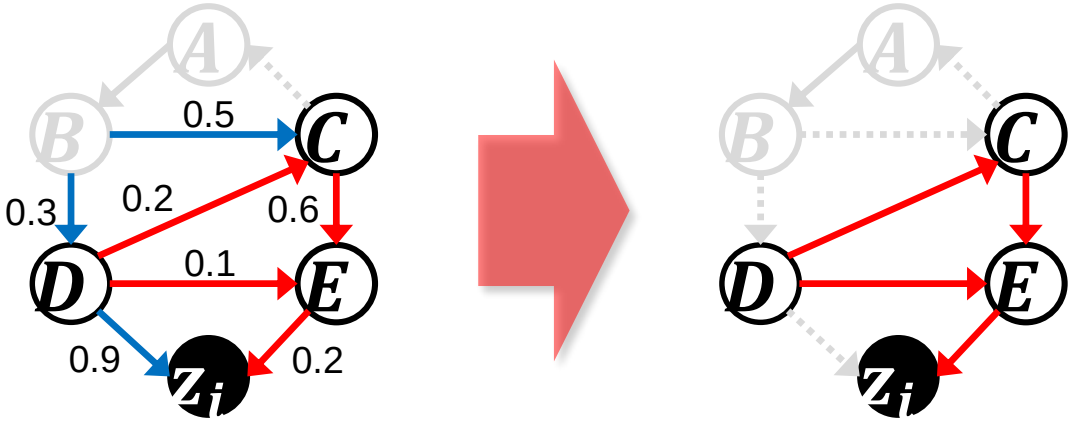
- 動的グラフ上の影響解析の索引手法を提案
  任意の時点でクエリの精度を保証
- 実験結果
  頂点・辺追加：**1秒未満**
  頂点・辺削除：**数十秒**
  影響力推定：**100,000頂点／秒**
  影響最大化：**50頂点抽出／秒**

今後の課題

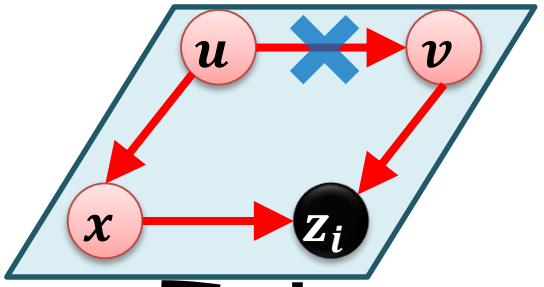
より大きいグラフでより高速に処理するには？

進行中の解決方策

- 索引構造の簡略化によるメモリ使用量の削減



- (最短路)木の更新による頂点・辺削除の高速化
  - 逆幅優先探索なく代わりの経路を見つける
    グラフ全体を走査する無駄な例 ⇨



ポスター発表で議論しましょう！

6/1(月) 9:00~11:00