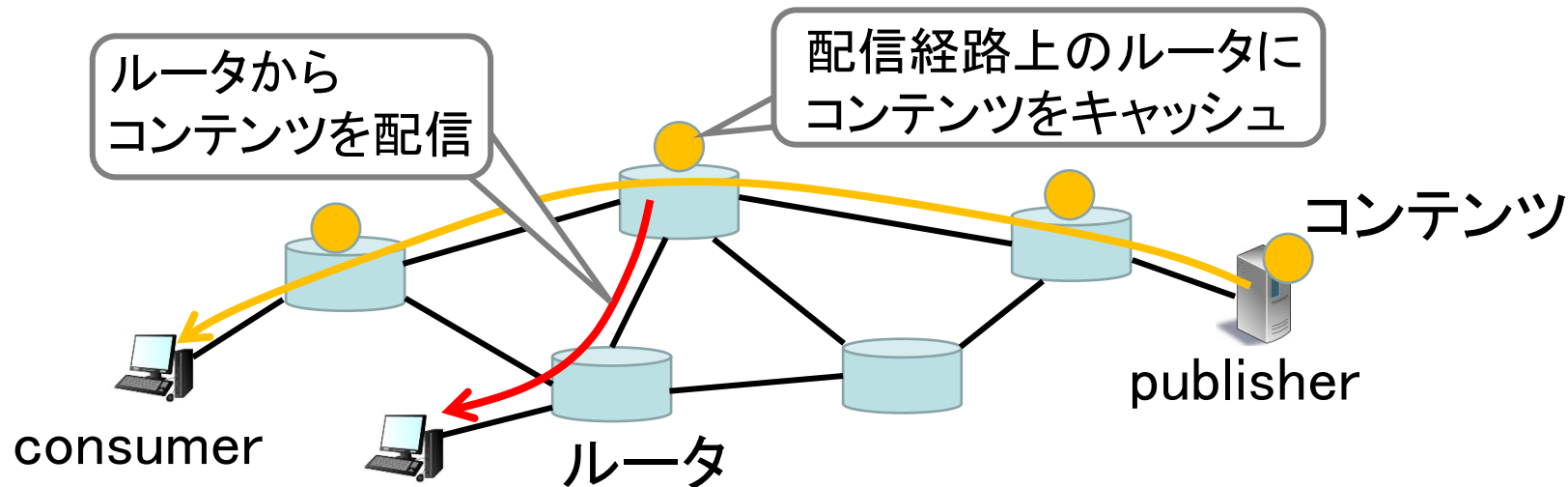

IOTAを用いた ICNの名称管理方式

立命館大学

岡田鉄平 上山憲昭

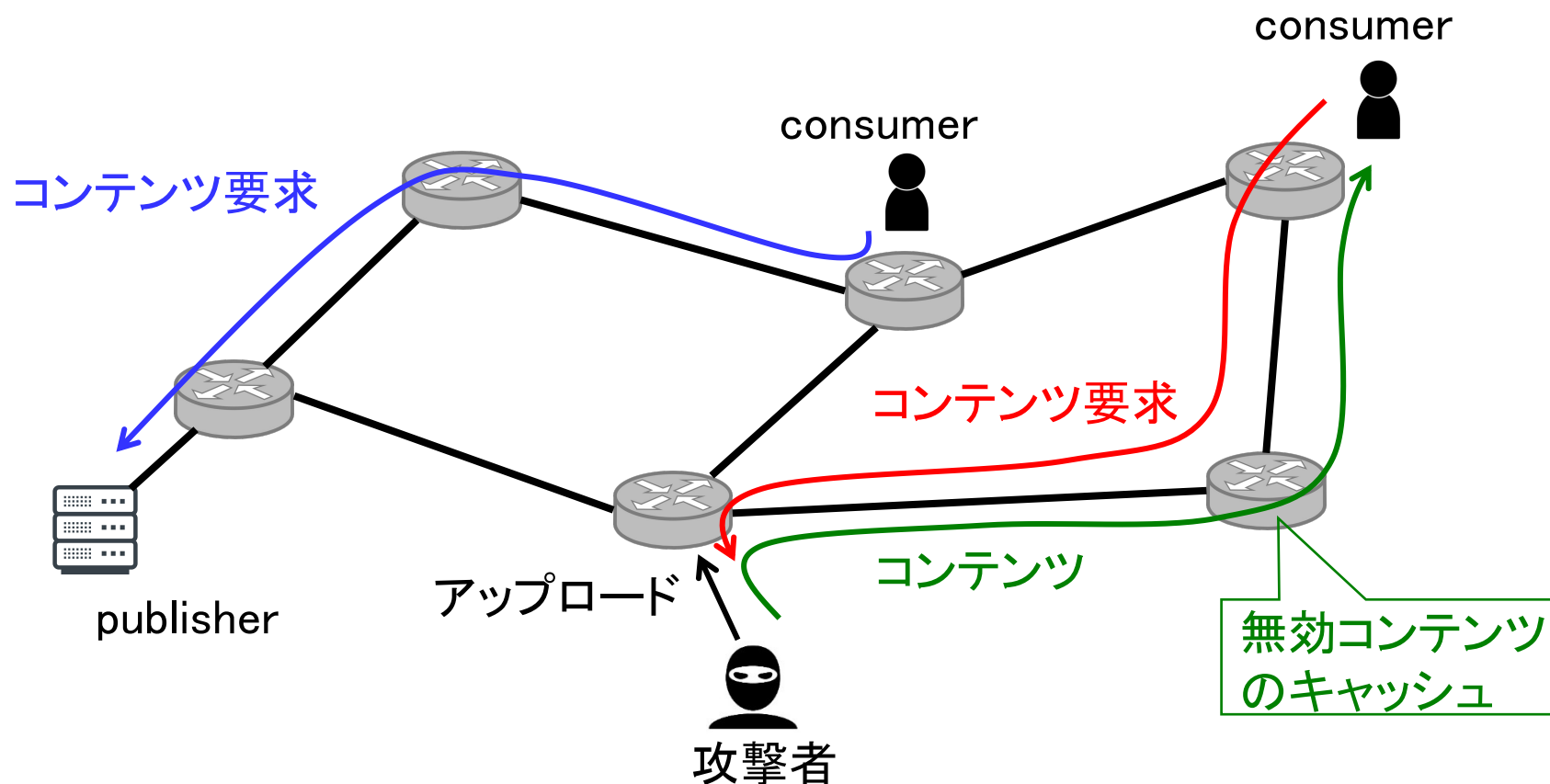
研究背景 (1/3)

- 情報指向ネットワーク(ICN: information-centric networking)
 - 要求されたコンテンツの名称をもとに配信者(publisher)からコンテンツ要求者(consumer)にコンテンツを転送
 - 経由するルータにコンテンツをキャッシュしながら配信



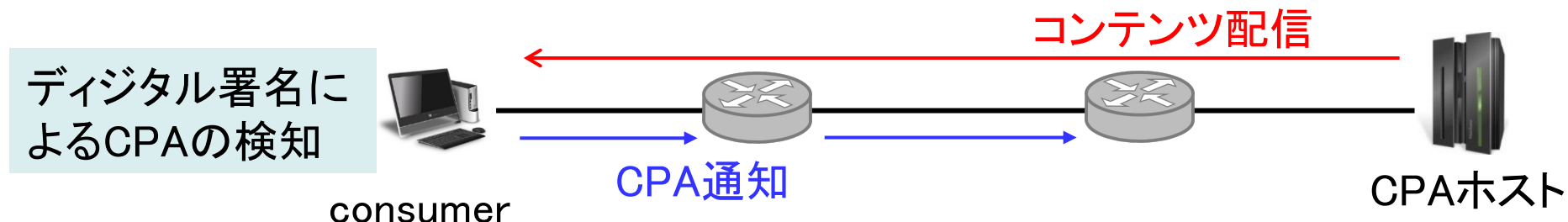
研究背景 (2/3)

- ICNでは、誰もがpublisherとしてコンテンツをアップロード可能
- 正当なpublisherを騙る攻撃者が、実在するコンテンツ名でfakeコンテンツをアップロードし、キャッシュの機能を低下→CPA (content poisoning attack)



研究背景 (3/3)

- consumerが公開鍵暗号を用いたデジタル署名によりコンテンツの正当性を判断し、不当コンテンツをルータに通知※1



- コンテンツと紐づいた公開鍵から生成されたデジタル署名と一致する偽のコンテンツをキャッシュに注入するfake型CPAは検知が困難
- 実在する高人気コンテンツを騙るfakeコンテンツをキャッシュに注入する詐称fake型CPAは対策が困難
 - 公開鍵を管理する認証局の職員が攻撃者と結託し、攻撃者の公開鍵に書き換える
 - アクセス数が多い人気コンテンツが詐称されると、攻撃の影響が大

※1: W. Cui, et al., “Feedback-Based Content Poisoning Mitigation in Named Data Networking”, IEEE ISCC 2018.

研究目的

- 詐称fake型CPA: 認証局のような一つの機関のみでデータを管理することが問題



- 登録データの改ざんが困難な分散型台帳技術の一つであるIOTAでコンテンツ名を管理するシステムを提案
- IOTA上での検索時間とメモリ量の増加が懸念→四つの検索方式間での性能評価

IOTA

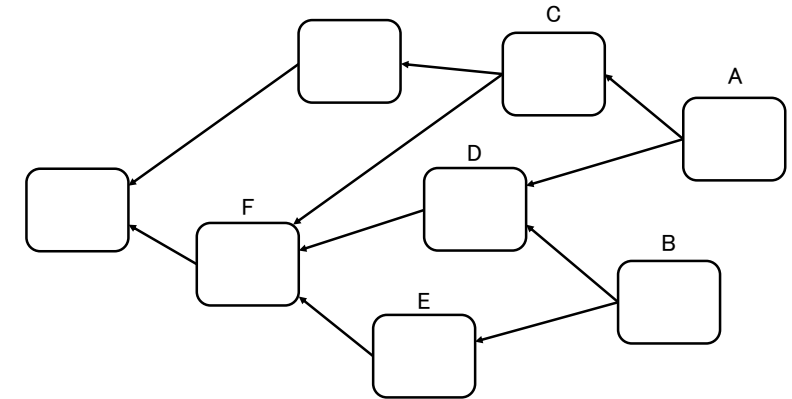
■ 分散型台帳技術の一つ

■ blockchain

- データ(transaction)をブロックにまとめ、ブロック毎に管理
 - ブロックサイズの制限→処理速度の低下による遅延の増大
- 承認作業には膨大な量の計算による電力が必要(PoW: proof of work)

■ IOTA

- transaction毎に管理→高速な処理が可能
- blockchainほどの計算量を必要としない



■ 有向非巡回グラフ(DAG: directed acyclic graph)構造

- 新しいtransactionが未承認のtransactionであるtipから二つ選択

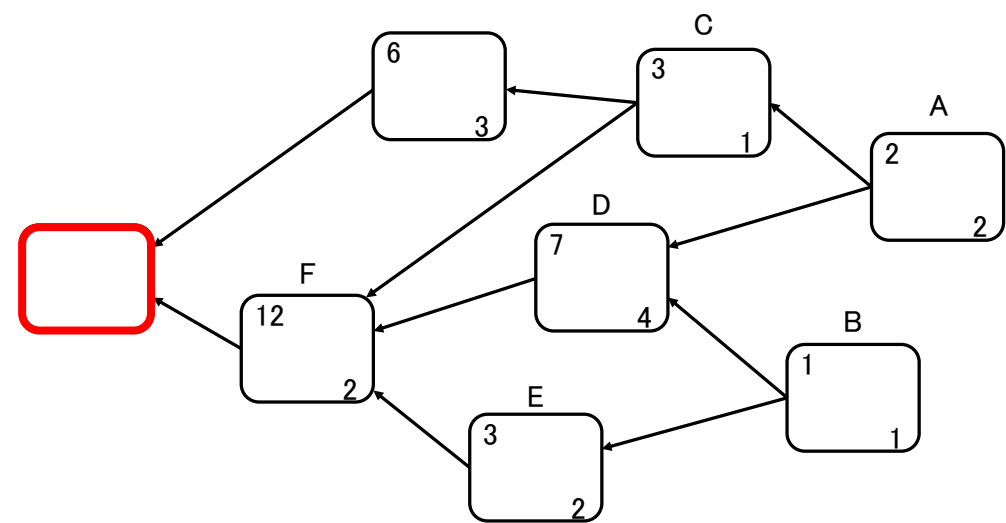
■ IOTAノードがクライアントからtransactionを受け取り、隣接するノードに伝播

tip選択アルゴリズム

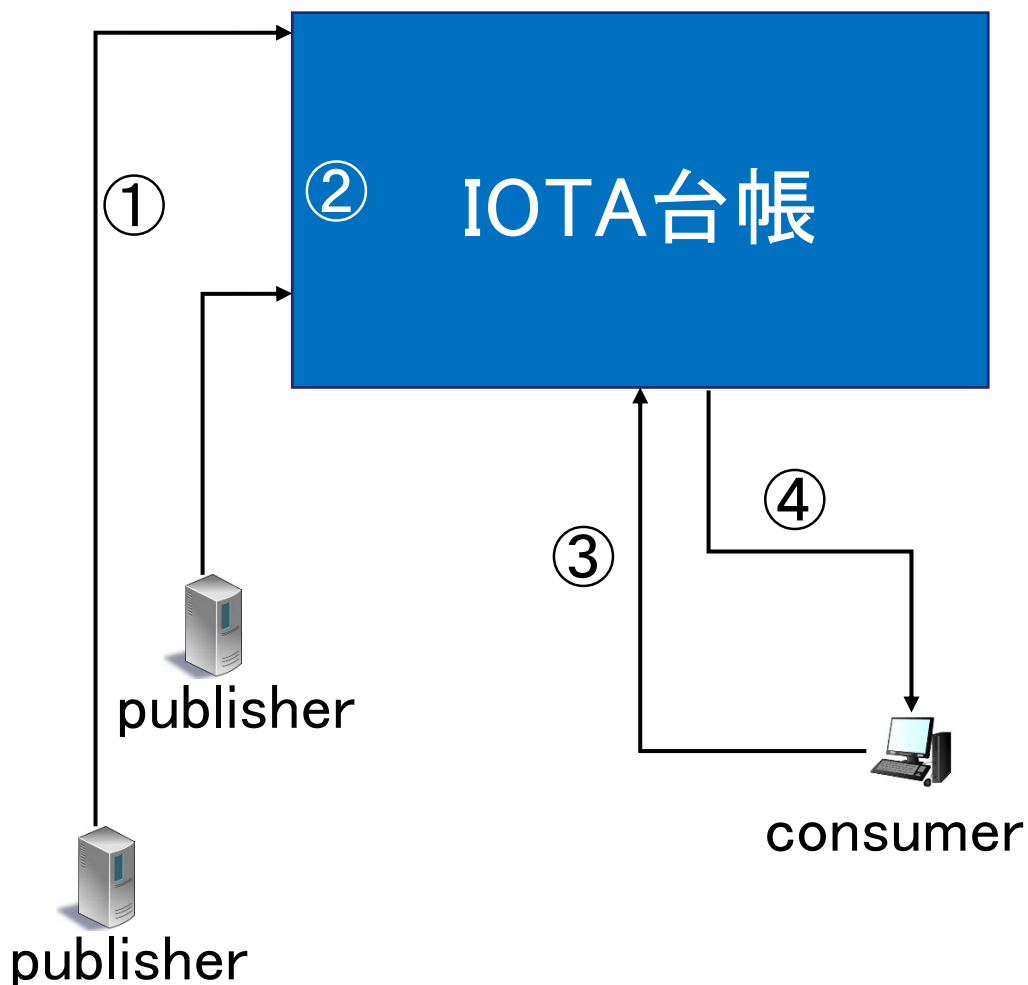
- 一様ランダム(URS: uniform random selection)
 - tipの中からランダムに二つ選択
- 重みなしランダムウォーク(URW: uniform random walk)
 - 最初のtransaction (ジェネシスtransaction)から等確率にtipを選択
- 重みありランダムウォーク(WRW: weighted random walk)
 - transactionの重みを考慮してtipを選択
 - transaction y から x への遷移確率 P_{xy}

$$P_{xy} = \frac{e^{-\alpha(H_x - H_y)}}{\sum_{z: z \rightarrow x} e^{-\alpha(H_x - H_z)}}$$

H_x, H_y : transaction x, y の累積重み
 $\alpha (\geq 0)$: 累積重みのパラメタ



提案方式



- ① publisherがコンテンツのアップロードに際し、transactionをIOTA台帳に登録
 - コンテンツのprefix, ID, 公開鍵, デジタル署名, コンテンツ名(= prefix + 公開鍵 + デジタル署名)
- ② 重複するコンテンツ名の管理を防ぐため、コンテンツのprefixで台帳内を探索
 - 重複がある場合、登録を拒否し、なければ登録
- ③ consumerがコンテンツのprefixを要求し、コンテンツ名を台帳内で探索
- ④ 発見したコンテンツ名をconsumerに回答
 - そのコンテンツ名で要求パケットであるinterestを送信し、コンテンツを要求

各コンテンツの正当なpublisherをIOTA上で管理

DAGのtransactionの検索法

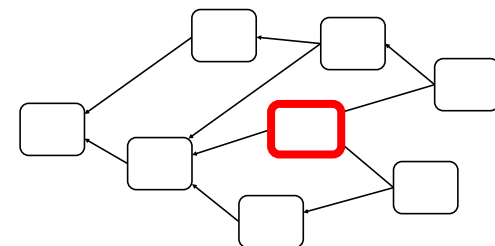
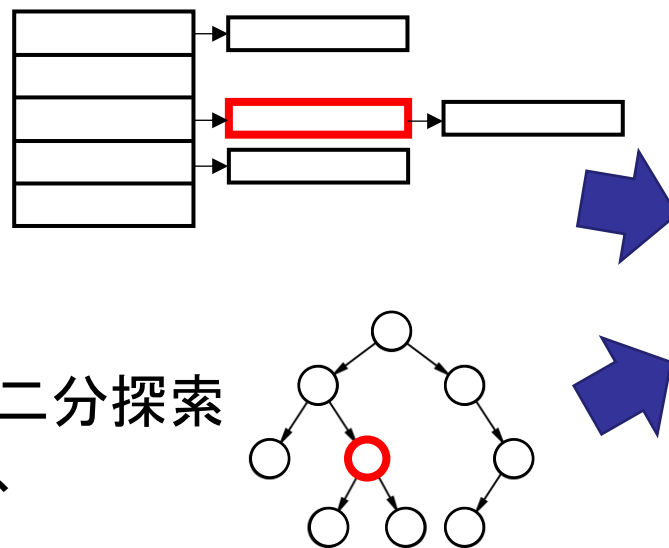
■ コンテンツ名を検索するタイミングは二回

- publisherによるコンテンツ名登録時
- consumerによる名前解決時

■ 四つの探索方式

- ハッシュチェーン法(hash)
- 二分探索木(bst: binary search tree)

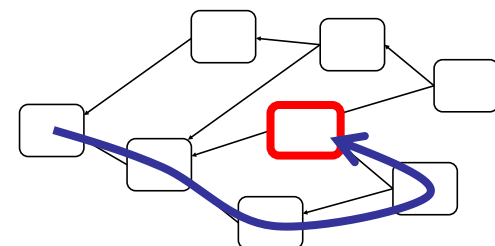
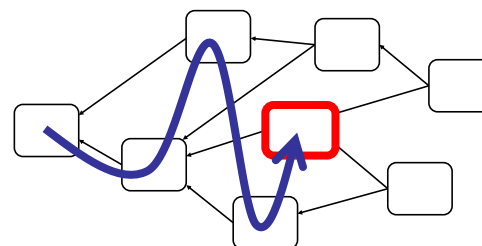
➡ prefixとIDをハッシュテーブルや二分探索木で管理→DAGに直接アクセス



- 幅優先探索(bfs: breadth-first search)

- 深さ優先探索(dfs: depth-first search)

➡ transactionにコンテンツ名が管理→発見後、探索終了



性能評価

- 計算機シミュレーションにより評価
 - 検索時間の実測値の平均値, 中央値, 上位95%値
 - 必要メモリ量

シミュレーション条件

URS: 一様ランダム
URW: 重みなしランダムウォーク
WRW: 重みありランダムウォーク

■ コンテンツ名登録時(transaction生成時)

■ コンテンツの名称のセット

- Alexaのwebページで公開されているアクセス数上位の8,000のwebページ※1を閲覧した際、エラーとならずに表示された7,131個のドメイン数

■ transaction数 N_t : 100, 1,000, 7,131 (URS, URW), 100, 1,000 (WRW)

- WRWにおいて, $N_t=7,131$ のとき処理が未終了

■ 遷移確率 P_{xy} のパラメタ α (重み): 0.1

■ 1秒あたりのtransaction生成数: 50

■ ハッシュチェーン法のテーブルのバケット数 B : 100

$$P_{xy} = \frac{e^{-\alpha(H_x - H_y)}}{\sum_{z: z \rightarrow x} e^{-\alpha(H_x - H_z)}}$$

■ consumerによる要求

■ 要求数: 5,000

■ 1秒あたりの要求発生数: 50

※1: Alexa webpage, <https://www.alexacom/siteinfo>.

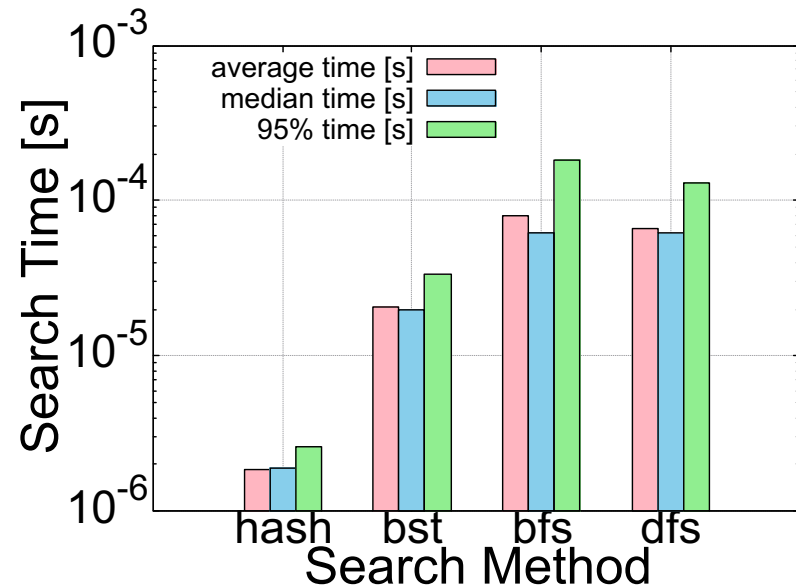
コンテンツ名登録時の検索時間

各探索方式の検索時間 ($N_t=100$)

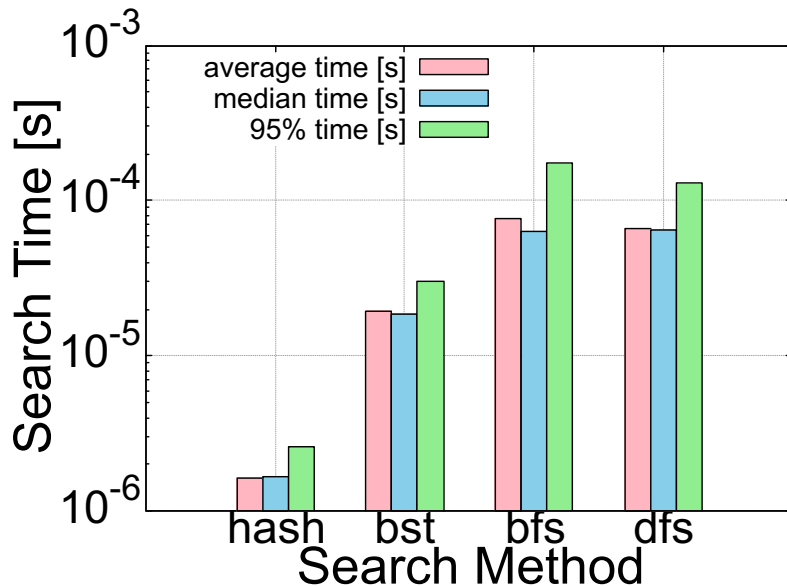
URS: 一様ランダム

URW: 重みなしランダムウォーク

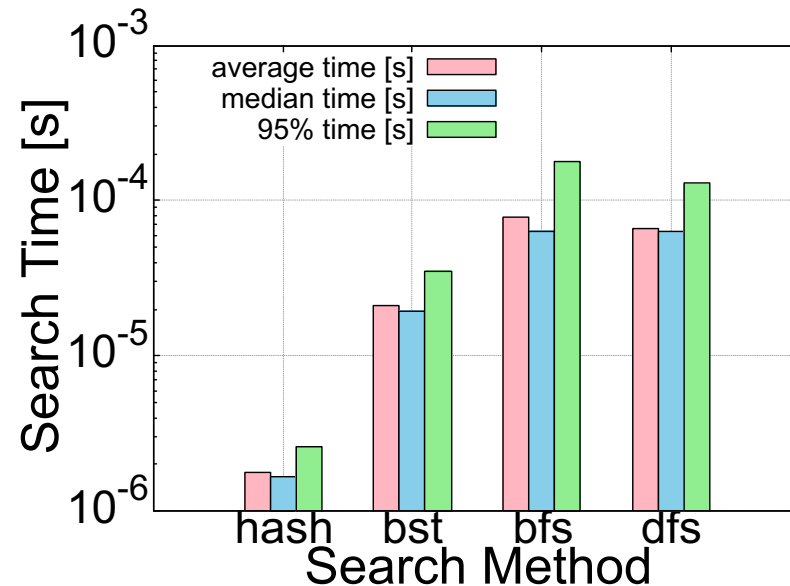
WRW: 重みありランダムウォーク



(a) URS



(b) URW



(c) WRW

■ ハッシュチェーン法 < 二分探索木 < 深さ優先探索 < 幅優先探索

- hash, bstでは, 必要な部分のみ探索するため, 検索時間が短

- bfs, dfsでは, しらみ潰しにDAGを全探索するため, 検索時間が長

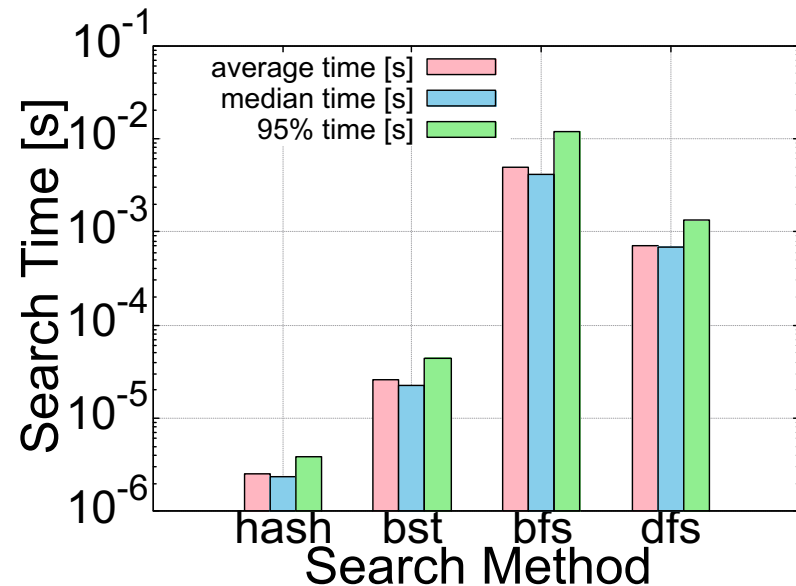
■ tip選択アルゴリズムごとに比較すると, どの手法も検索時間の差はほとんど無

- hash, bst: DAGとは別のテーブルで管理→DAGの形状に影響を受けない

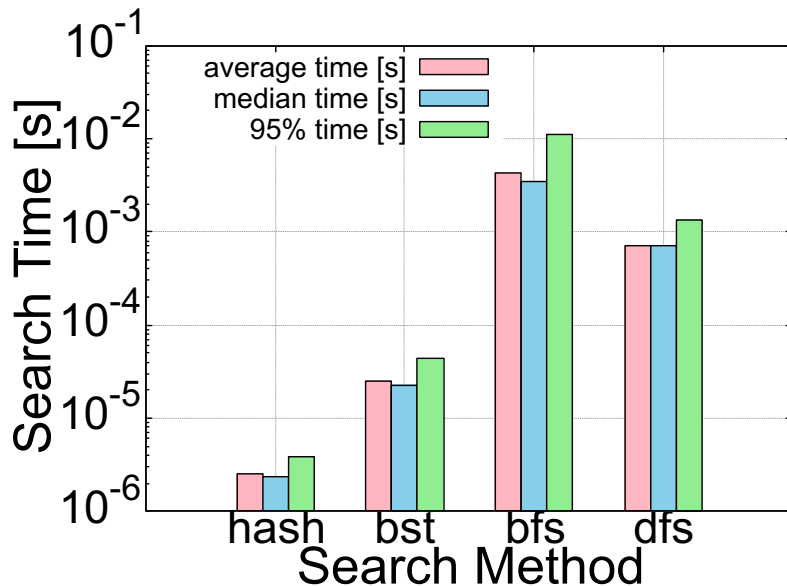
- bfs, dfs: 少ないtransaction数では, DAGの形状に差がほとんど無

各探索方式の検索時間 ($N_t=1,000$)

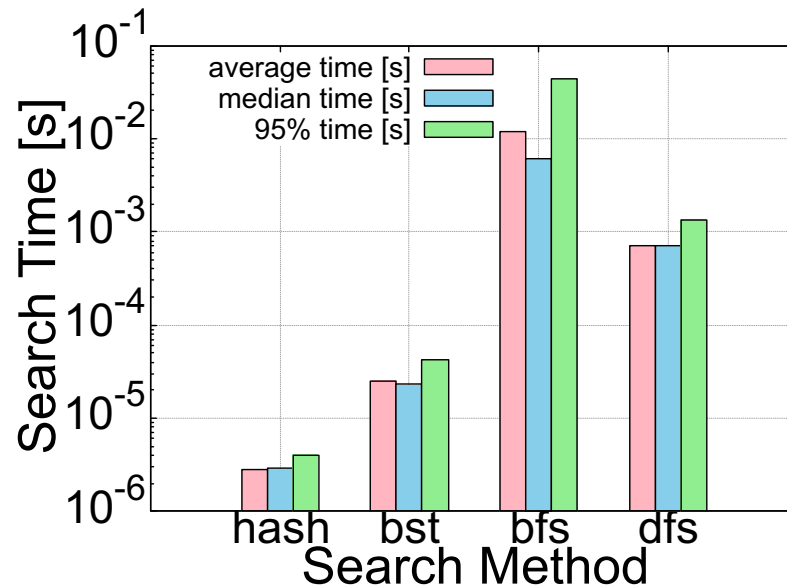
URS: 一様ランダム
URW: 重みなしランダムウォーク
WRW: 重みありランダムウォーク



(a) URS



(b) URW

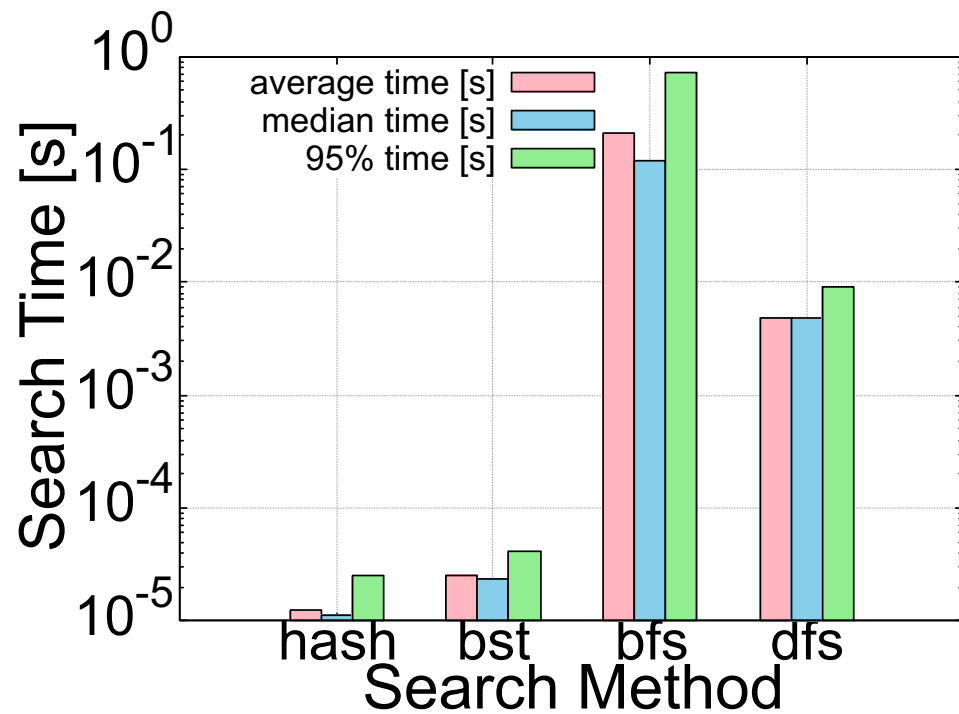


(c) WRW

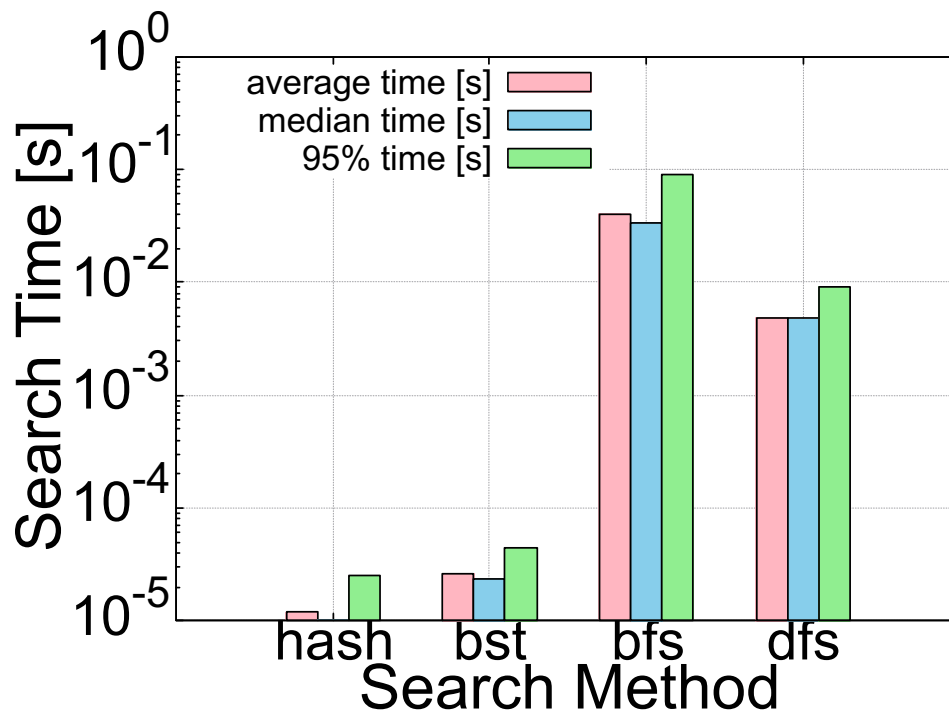
- $N_t=100$ と比較すると, DAGの規模が大きくなるため, hashおよびbstと, bfsおよびdfsの差が顕著
- WRWのbfsに注目すると, URS, URWと比べて検索時間が大
 - WRW: tipの重みが大きいtransactionが優先的に選択
 - ジェネシスtransactionからのホップ数大きなtransactionが多

各探索方式の検索時間 ($N_t=7,131$)

URS: 一様ランダム
URW: 重みなしランダムウォーク



(a) URS



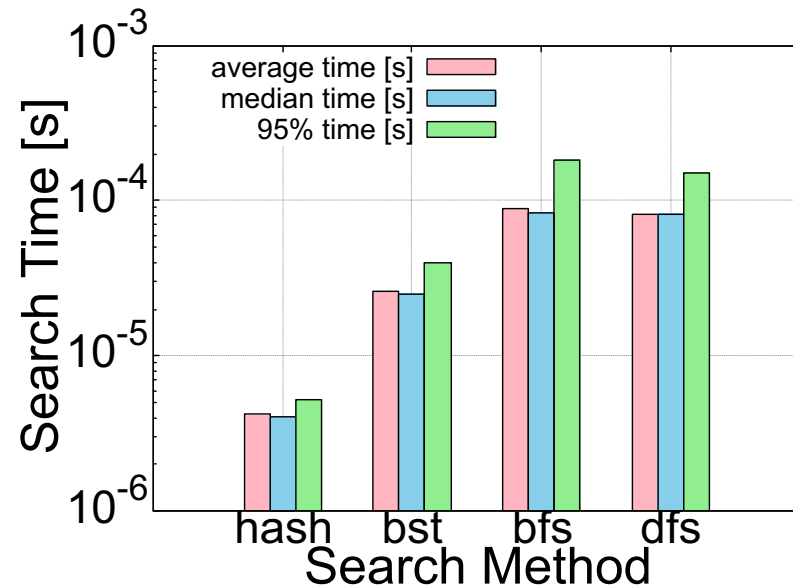
(b) URW

- bfsにおいて, URSの方がURWより検索時間が大
 - URS: tipの中からランダムに選択→ホップ数の大きなtransactionでも選択
 - URW: ジェネシスtransactionから等確率に選択→ホップ数の小さなtransactionがDAGに多数存在

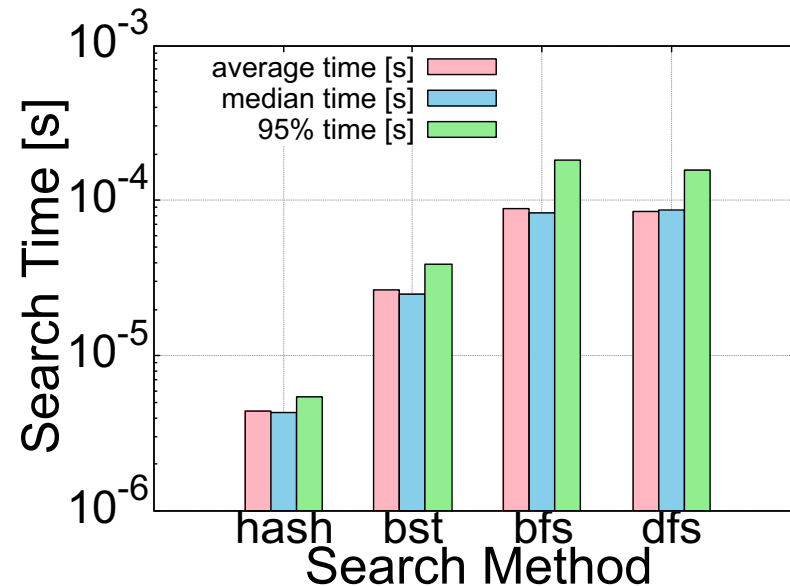
名前解決時の検索時間

各探索方式の検索時間 ($N_t=100$)

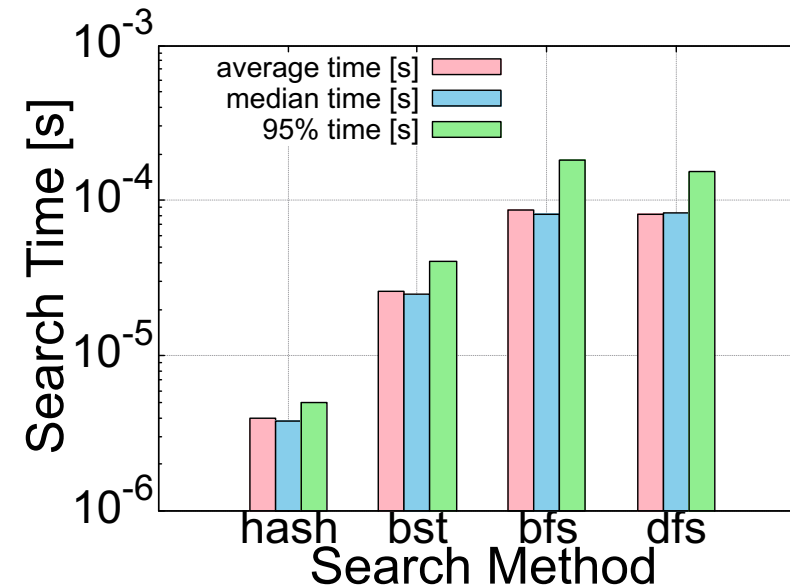
URS: 一様ランダム
URW: 重みなしランダムウォーク
WRW: 重みありランダムウォーク



(a) URS



(b) URW

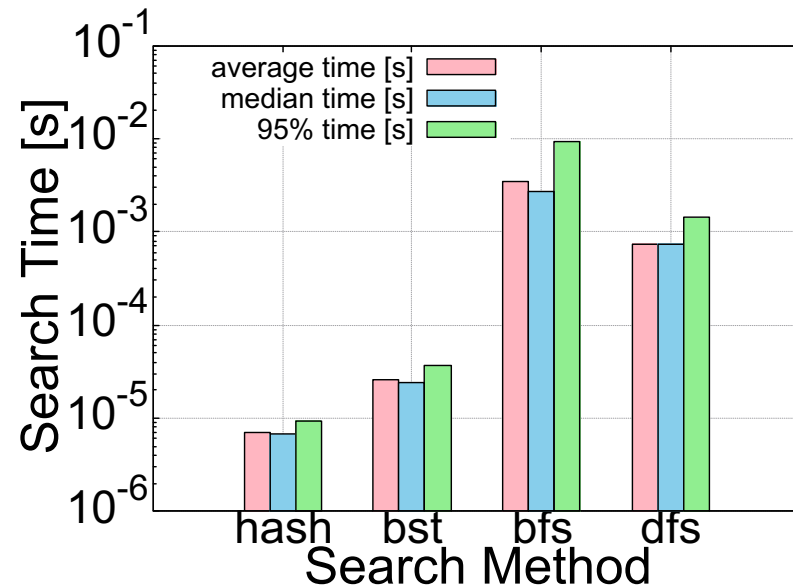


(c) WRW

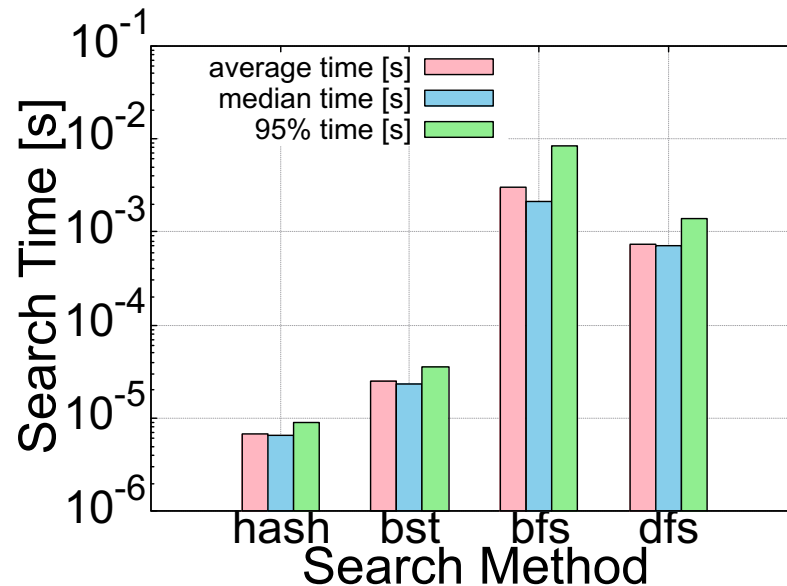
- ハッシュチェーン法 < 二分探索木 < 深さ優先探索 < 幅優先探索
- transaction数が少ないため, tip選択アルゴリズムによる検索時間の差は小
 - コンテンツ名登録時の $N_t=100$ と比較しても, 差はほとんど無

各探索方式の検索時間 ($N_t=1,000$)

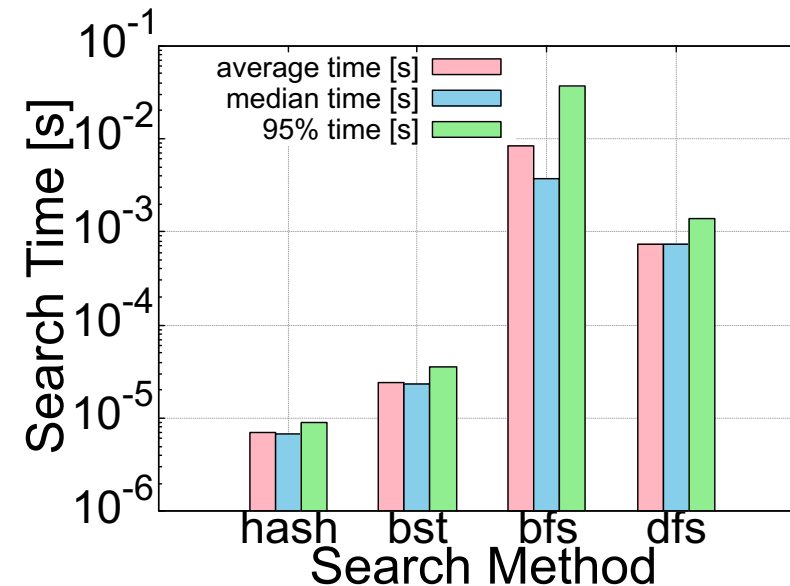
URS: 一様ランダム
URW: 重みなしランダムウォーク
WRW: 重みありランダムウォーク



(a) URS



(b) URW

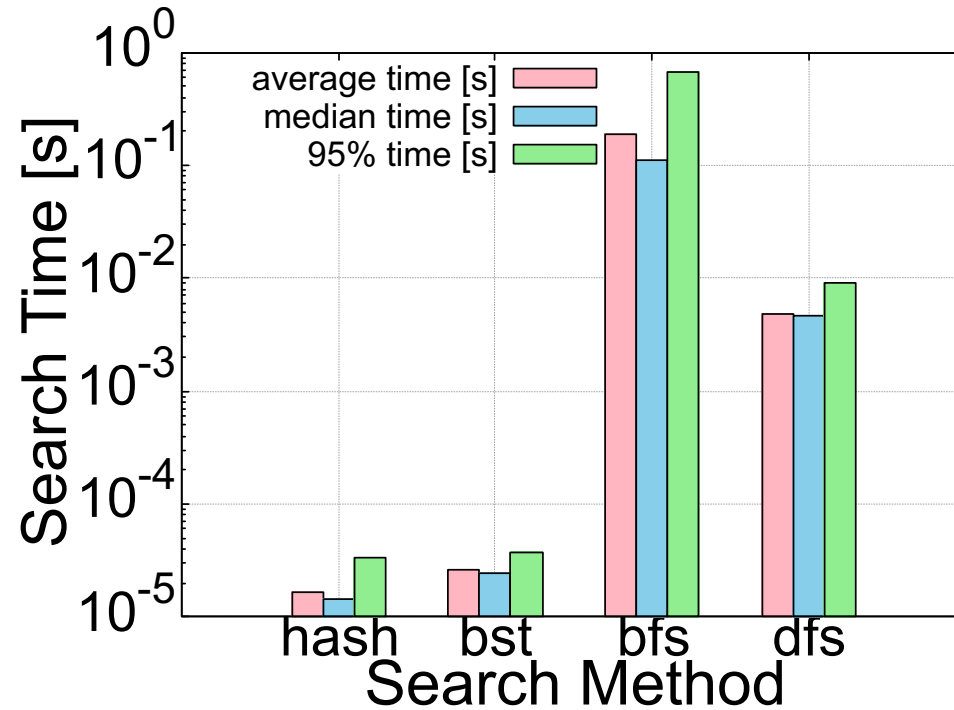


(c) WRW

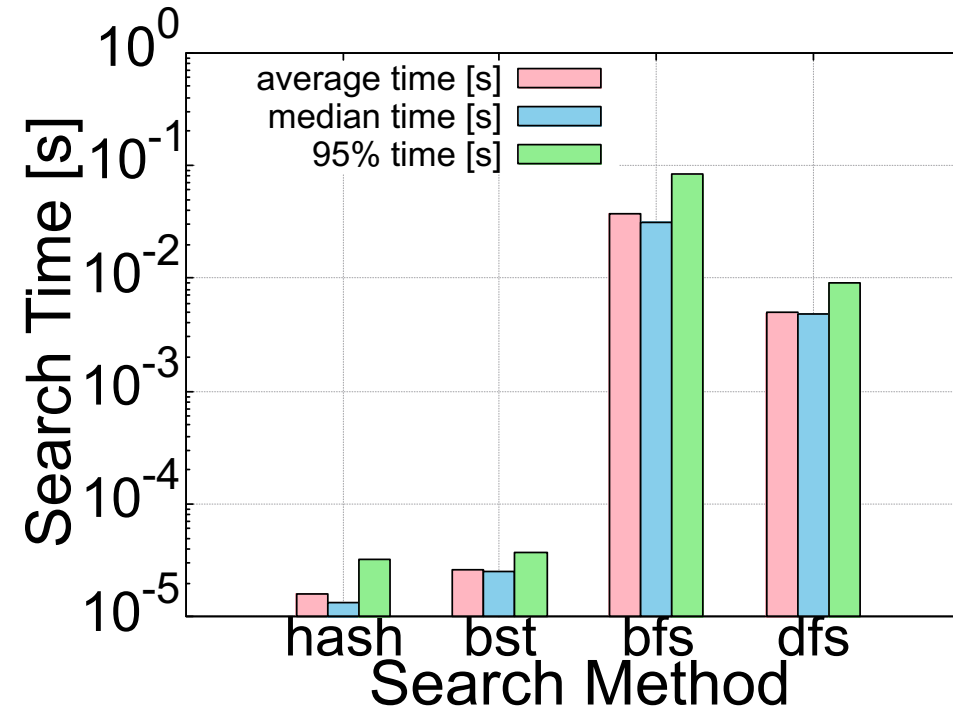
- テーブルなどで管理するhashおよびbstと、DAGで直接管理するbfsおよびdfsとの差が大

各探索方式の検索時間 ($N_t=7,131$)

URS: 一様ランダム
URW: 重みなしランダムウォーク



(a) URS

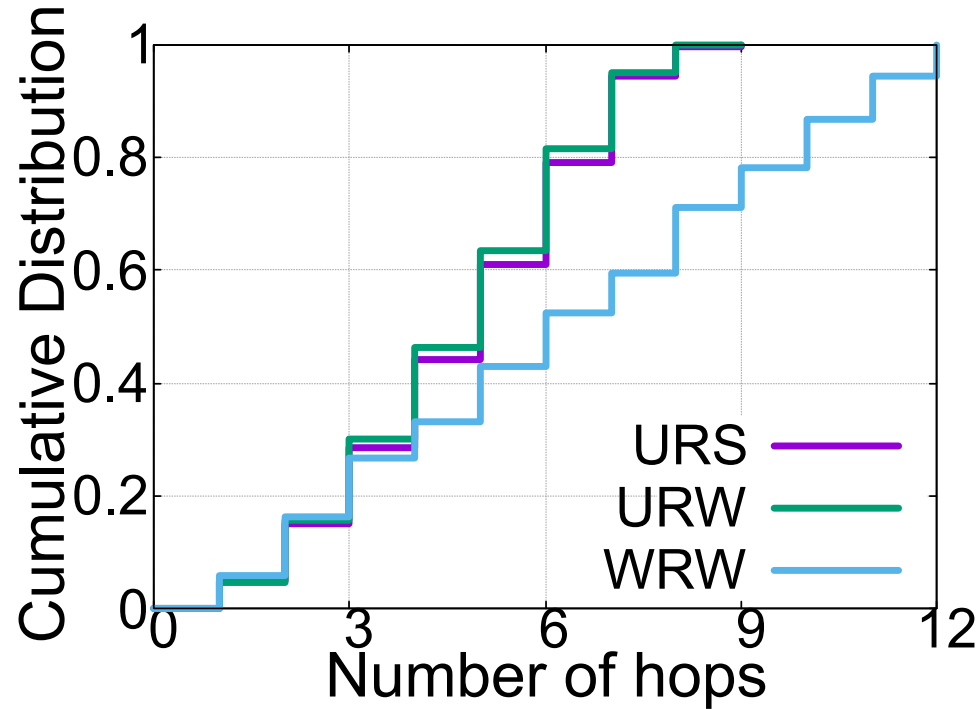


(b) URW

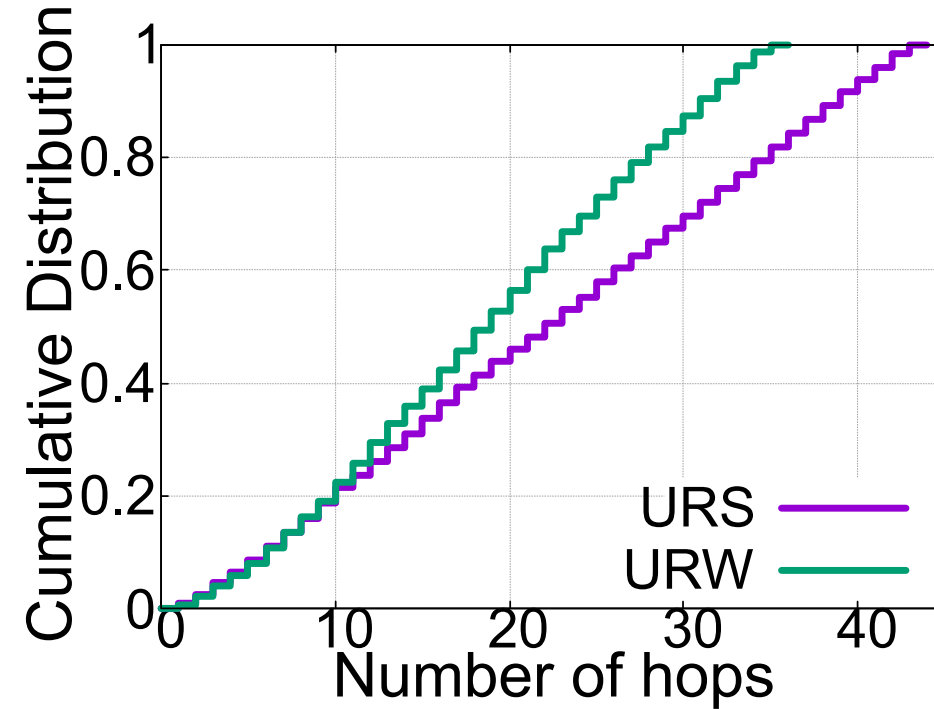
■ bfsにおいて, $URS > URW$

DAG上のtransactionのジェネシスtransactionからのホップ数の分布

URS: 一様ランダム, URW: 重みなしランダムウォーク, WRW: 重みありランダムウォーク



(a) $N_t=1,000$

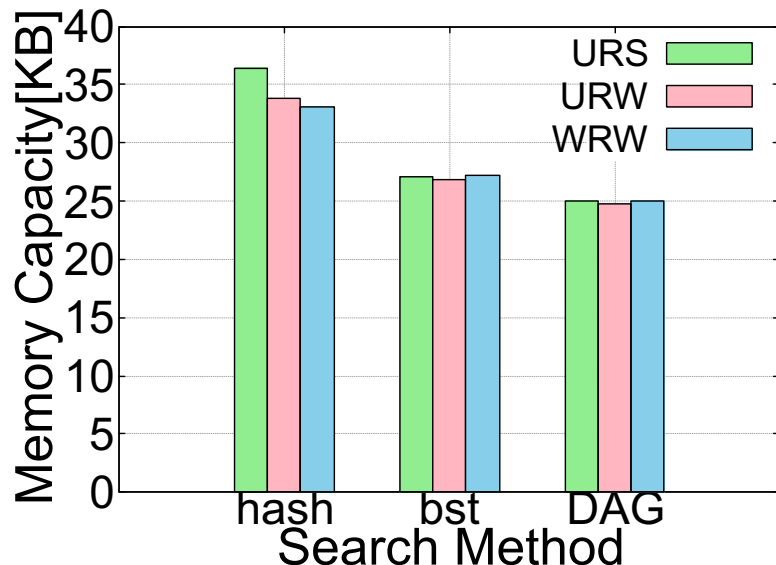


(b) $N_t=7,131$

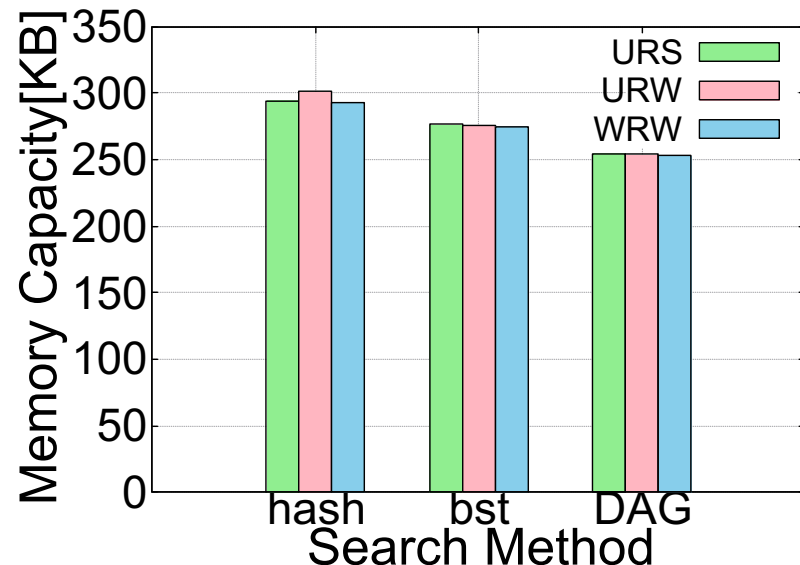
- $N_t=1,000$ では, WRWがホップ数の大きなtransactionが他二つと比較して多
 - 幅優先探索で, WRWが探索に最も時間を費やす
- $N_t=7,131$ では, URSの方が, URWよりもホップ数の多いtransactionが多
 - 幅優先探索では, $URS > URW$

各探索方式の必要メモリ量

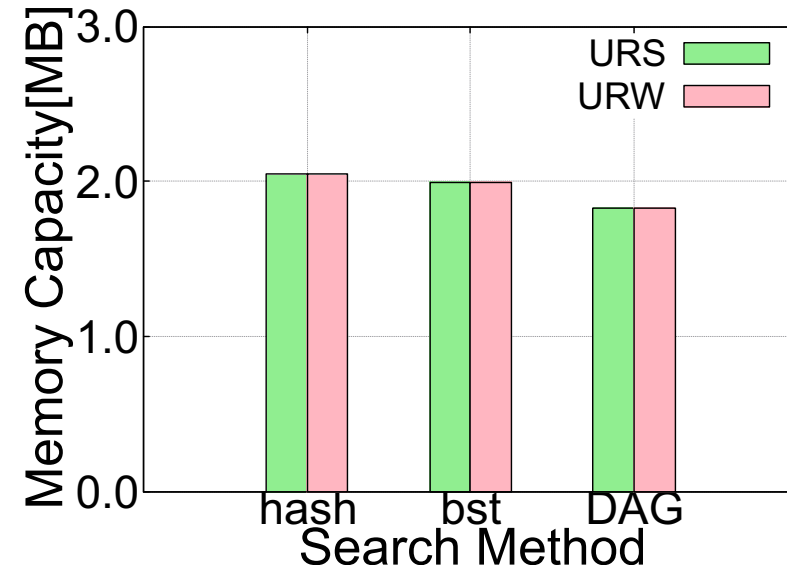
URS: 一様ランダム
URW: 重みなしランダムウォーク
WRW: 重みありランダムウォーク



(a) $N_t=100$



(b) $N_t=1,000$



(c) $N_t=7,131$

- ハッシュチェーン法 > 二分探索木 > DAG
- hash: 最大容量を持つバケットのメモリ量が, 全てのバケットで確保
 - 未使用メモリ多
- bst: ノード数分のメモリを確保
- 検索時間と必要メモリ量のトレードオフの関係を確認

まとめ・今後の方針

- publisherによるコンテンツのアップロードに際し, IOTAでコンテンツ名を管理
- 詐称fake型CPAを未然に防ぐ方式を提案
- 台帳内でコンテンツ名を検索する四つの探索手法の検索時間と必要メモリ量の比較
 - 検索時間: ハッシュチェーン法 < 二分探索木 < 深さ優先探索 < 幅優先探索
 - 必要メモリ量: ハッシュチェーン法 > 二分探索木 > DAG
 - 検索時間と必要メモリ量のトレードオフの関係を確認
- 今後の方針
 - transaction数の増加に伴う, IOTAのスケール性の評価