令和4年度　春学期　卒業研究3 (BI)
学士論文

題目 ICNにおけるIOTAを用いたコンテンツ名管理方式

指導教員 上山憲昭 教授

立命館大学 情報理工学部 セキュリティ・ネットワークコース

学籍番号 26001900764
岡田鉄平

令和5年1月31日

概要

従来のインターネットのように，IP アドレスをもとにコンテンツを転送するのではなく，要求されたコンテンツの名称をもとにルータにキャッシュしながらコンテンツ要求者 (Consumer) に配信を行う情報指向ネットワーク (ICN: information-centric networking) が次世代のネットワークとして注目を集めている．この ICN において，誰でも Publisher としてコンテンツをアップロードすることが可能だが，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することでキャッシュの機能を低下させる CPA (content poisoning attack) の問題が指摘されている．既存研究では，CPA の対策として，Consumer が受信したコンテンツの正当性を，公開鍵を用いたディジタル署名によって判断し，不当なコンテンツをルータに通知する方式が提案されている．しかし，コンテンツに紐づいた公開鍵により生成されたディジタル署名と一致する fake 型 CPA では，先の方式では検知は困難となる．中でも，公開鍵を管理している認証局 (CA: certification authority) の職員が攻撃者と結託し，正当な Publisher の公開鍵を攻撃者のものと入れ替えることにより，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 は対策が困難である．そこで本論では，Publisher によるコンテンツのアップロードに際し，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し，詐称 fake 型 CPA を未然に防ぐ方式を提案する．提案方式では，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を行った．その結果，検索時間と必要メモリ量のトレードオフの関係を確認した．

目 次

第1章　序論

1.1　研究の背景

従来のインターネットでは，ユーザがドメインネームの名前解決を DNS (domain name system) サーバに依頼し，そこで回答された IP アドレスに基づいて動画像やウェブサイトなどのコンテンツの配信を行う．一方，インターネットによるデータの要求は，DNS サーバの名前解決に時間がかかり，負荷が大きくなることが懸念される．そこで，通信開始時に名前解決を行わず，要求されたコンテンツの名称をもとに要求パケット (Interest) を転送し，配信サーバ (Publisher) からコンテンツ要求者 (Consumer) へ経由するルータにキャッシュしながらコンテンツを配信する情報指向ネットワーク (ICN: information-centric networking) が，次世代のコンテンツ配信方式として注目を集めている．

この ICN において，Consumer は取得したいコンテンツの名称を Interest として送信する．その Interest を受信したルータのキャッシュに要求コンテンツが存在する場合は，Interest を棄却し，そこからコンテンツの配信を行い，存在しない場合は，要求コンテンツが存在するルータ，もしくは Publisher に Interest を転送し，同様に配信を行う．配信元から経由するルータにキャッシュしながらコンテンツを転送するため，次回 Consumer から同じ要求があった場合，そのコンテンツがキャッシュされている近くのルータから配信を行うことが可能であり，配信コストの抑制が期待される．

ICN では誰もが Publisher としてコンテンツをアップロード可能だが，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することでキャッシュの機能を低下させる CPA (content poisoning attack) の問題が指摘されている [1]．CPA の対策として，Consumer が公開鍵暗号を用いたディジタル署名によりコンテンツの正当性を判断し，不当なコンテンツをルータに通知する方式が提案されている [2]．CPA は，コンテンツに紐づいた公開鍵により生成されたディジタル署名と一致する fake 型と，一致しない corrupted 型があるが [3]，前者に対しては [2] の方式では検知は困難となる．中でも，公開鍵を管理する認証局 (CA: certification authority) の職員が攻撃者と結託し，攻撃者の公開鍵と書き換えることにより，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，対策が困難である．本攻撃は，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きいことが推測される [4]．また，正当な Publisher の公開鍵を乗っ取って書き換えているため，攻撃者のコンテンツの方が正当化される．

1.2 研究の目的

詐称 fake 型 CPA は，公開鍵を CA のような一つの機関のみで管理することにより発生する．

そこで本論では，分散型台帳技術の一つである IOTA[5] を用いてコンテンツ名を管理することで詐称 fake 型 CPA を未然に防ぐ方式を提案する．分散型台帳としてはが代表的であるが，はスケール性に課題があるのに対し，IOTA はスケール性が高い．また，台帳内でコンテンツ名を探索するための手法として，ハッシュチェイン法，二分探索木，幅優先探索，深さ優先探索の検索時間と必要メモリ量を比較する．

第2章　関連研究

2.1　CPA

ICN において，偽のコンテンツをルータに注入することでキャッシュの機能を低下させる CPA には，fake 型と corrupted 型がある [3]．fake 型ではコンテンツと紐づいた公開鍵により署名されたディジタル署名と一致する fake コンテンツをルータに注入し，corrupted 型ではディジタル署名と一致しないコンテンツをルータに注入する．ゆえに，corrupted 型 CPA に対しては署名を検査することにより容易に検知できるが，fake 型 CPA では検知が困難である．さらに [4] では，fake 型を独自 fake 型および詐称 fake 型に分類している．

独自 fake 型では，攻撃者が独自に作成した架空のコンテンツを結託ユーザから要求することで，ルータに注入する．この手法では正常ユーザからの要求はなく，攻撃者と少数の結託ユーザからの要求であるため，偽のコンテンツがネットワークを流れる量は少ない．ゆえに，fake コンテンツがルータにキャッシュされるのは一時的であるため，攻撃の影響は小さいと推測される．

詐称 fake 型では，実在するコンテンツを騙る偽のコンテンツを正常ユーザから要求することで，ルータに注入する．この手法では攻撃者が正当なコンテンツを乗っ取って偽のコンテンツを配信する．また，多数の正常ユーザから要求があるため，長時間キャッシュに残る可能性が高く，攻撃の影響は高いと推測される．

本論では特に，攻撃の影響が大きい詐称 fake 型 CPA を防ぐことに焦点を当てている．

2.2　IOTA

分散型台帳の例として Blockchain が挙げられるが，Blockchain では transaction と呼ばれるデータをブロックごとで管理している．そのため，一つのブロックに格納できる transaction 数に制限がある．また，trasnaction の承認にかかる手数料や，遅延時間が長いといったデメリットが存在する．一方，IOTA では，transaction ごとに管理しているため，Blockchain で挙げられているスケーラビリティの問題を考慮する必要がない．また，手数料が無料であることも IOTA の特徴の一つである．[6] では，エネルギー市場において任意の家庭同士が電力会社などを介さず，IOTA 台帳を通して直接電力を売買する方式が提案されている．

2.3　IOTA-VPKI

協調高度道路交通システム (C-ITS: cooperative intelligent transport system) により，車両に搭載する OBUs (on-board units) と道路に設置される RSUs (roadside units) によっ

て，車車間通信や路車間通信などの V2X (vehicle to x) 通信を可能にするネットワークが構成される．これにより，事故や渋滞などの課題の解決が期待される．一方で，通信内容の盗聴や改ざんなど，セキュリティに関する課題にも対処する必要がある．CA ベースの車両用公開鍵基盤 (VPKI: vehicular public key infrastructure) を活用することで，車両は秘密鍵で署名された証明書付きのメッセージを送信することができるが，毎回同じ秘密鍵で署名すると，他者も同じ公開鍵で検証することができ，送信車両の行動や所在などの情報が特定される危険性がある．そこで，[8] では，プライバシー保護を重視した VPKI システムが提案されている．この方式では，検証に使われる秘密鍵，公開鍵を高頻度で変更することで，先のようなリスクが軽減される．

しかし，[8] の方式では，CA に単一障害点 (SPoF: single point of failure) があり，認証情報の不正使用の危険性がある．そこで [7] では，このシステムに IOTA 台帳を導入することで SPoF を排除し，証明書発行の透明性を保証する方式が提案されている．この方式の手順は，以下の通りである．

1. 送信車両はメッセージを送信する際，証明書を作成
2. CA は証明書のハッシュ値を CA の秘密鍵で署名したものを IOTA 台帳に登録し，それが管理されている台帳のアドレスを送信車両に返送
3. 送信車両は受け取ったアドレスをメッセージに添付し，受信者はそのアドレスをもとに台帳に直接アクセスして CA の公開鍵で検証．証明書のハッシュ値は公開されているため，検証の結果一致していればメッセージの正当性，信頼性を確認可能

この方式では，IOTA 台帳を用いたディジタル署名方式を実現している．また，証明書のハッシュ値のみが公開されているため，台帳上の情報を使って証明書を盗むことは困難である．

第3章　提案方式

3.1　IOTA

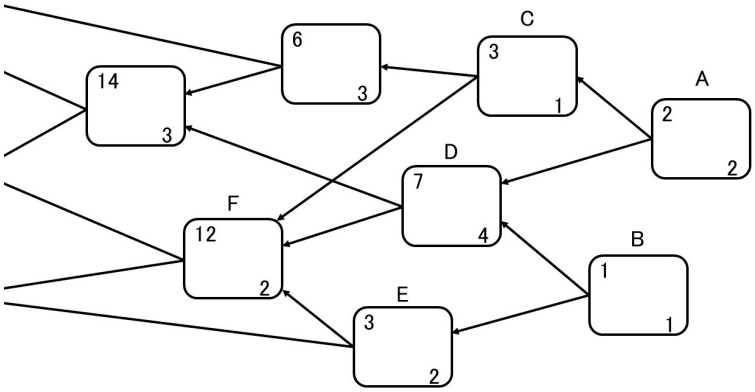

図 3.1: IOTA における DAG

IOTA では，新しい transaction が未承認の transaction である tip の中から二つ選択し，Tangle と呼ばれる有向非巡回グラフ (DAG: directed acyclic graph) を形成することで分散型台帳が実装される [5]．図 3.1 に DAG の図を示す．A, B, C, D, E, F は transaction を表しており，それぞれ既存の二つの transaction を参照する．また，右下の数字はその transaction の重みを，左上の数字は累積重みを表している．例えば，F の累積重みは 12 で，これは A, B, C, D, E の重みと自身の重みを足し合わせたものとなっている．また，選択されていない A や B のような transaction が tip である．その tip 選択アルゴリズムは以下に示す三種類である．

Uniform random selection (URS) では，存在している tip の中からランダムに二つ選択する．

Unweighted random walk (URW) では，最初の transaction であるジェネシス transaction から参照されている transaction を等確率に選択していき，tip を選択する．これを二回行うことで，二つの tip を選択する．

Weighted random walk (WRW) では，URW と同様にジェネシスノードから選択するが，累積重みを考慮して選択する．WRW では，重みの大きな transaction が優先して選択される．また，transaction y から x への遷移確率 P_{xy} は次式により得られる．

$$P_{xy} = \frac{e^{-\alpha(H_x - H_y)}}{\sum_{z:z\to x} e^{-\alpha(H_x - H_z)}} \tag{3.1}$$

ここで，H_x, H_y は，transaction x および y の累積重みを表している．また，$\alpha(\geq 0)$ は累積重みのパラメタである．$\alpha = 0$ のときは重みのないランダムウォーク，つまり URW となる．また，α の値が大きくなるにつれ，累積重みの影響が大きくなるため，選択される tip に偏りが生じる．IOTA では，同じ暗号資産を複数回使用することにより攻撃者が不正に資金を取得する二重支払い攻撃が問題視されている．その例として，攻撃者が Tangle を二手に分岐させて二重支払い transaction の累積重みを増やすことで，それを承認させる分裂攻撃 [9] や，メイン Tangle とは別の Tangle に二重支払い transaction を追加し，短時間でリンクの数を増やすことで承認させやすくするパラサイトチェーン攻撃 [10] が挙げられる．これらの攻撃は，tip 選択の際に累積重みを考慮しない URS や URW で発生する可能性が高いため，対策として Tangle に分岐や分裂が生じても累積重みの大きい方のみに遷移されるように，式 (3.1) の α の値を大きく設定した WRW を採用することが有効である．

提案方式では，transaction にコンテンツの Prefix，ID，公開鍵およびそれに対応するディジタル署名，そして Prefix，公開鍵，ディジタル署名により構成されたコンテンツ名を管理する．分散型台帳の性質上，登録データの改ざんは困難であるため，正当性が担保される．

3.2 提案方式の概要

以下に，提案方式の流れを示す．

1. Publisher がコンテンツのアップロードに際して，コンテンツの Prefix，ID，公開鍵，ディジタル署名，コンテンツ名を IOTA 台帳に登録
2. 重複するコンテンツ名の管理を防ぐため，既に同じコンテンツ名が管理されているか IOTA 台帳内で検索し，既に存在していれば登録を拒否し，そうでなければ登録
3. Publisher によるアップロードが終了した後，Consumer がコンテンツの Prefix を要求
4. 要求された Prefix で IOTA 台帳内を検索し，ヒットしたコンテンツ名を Consumer に回答
5. Consumer がそのコンテンツ名で Interest を送信し，コンテンツを要求

提案方式では，DNS の名前解決のような流れで Consumer がコンテンツ名を取得する．また，DAG の検索法として，四つの手法を次のセクションで示す．

3.3 DAG におけるコンテンツ名探索手法

提案方式では，DAG 内でコンテンツ名を検索するタイミングは，Publisher によるコンテンツ名登録時と，Consumer による名前解決時の二つである．本研究で着目した四つの探索手法を以下に示す．

ハッシュチェイン法では，コンテンツの Prefix を数値に変換し，ハッシュ関数にかけて算出したハッシュ値をもとに，ハッシュテーブルの各要素であるバケットにデータを格納する．異なるデータでも同じハッシュ値となる衝突が発生しうるが，ハッシュチェイン法ではそれらを連結リストで繋ぐことで，同じバケットに複数のデータを管理することが可能である．提案方式では，ハッシュテーブルでコンテンツの Prefix と ID を管理し，その ID をもとに DAG 上に直接アクセスすることでコンテンツ名を取得する．

二分探索木 (bst: binary search tree) は木構造の一種で,「左の子 < 親 < 右の子」という性質を持っている．最上位のノードである根から探索を開始し，探索対象がノードの値より小さければ左に，大きければ右に移動する．この流れを発見するまで繰り返す．二分探索木でもハッシュチェイン法と同様に，コンテンツの Prefix と ID を管理する．Prefix を数値に変換した後，二分探索木で発見した ID を参照して DAG 上に直接アクセスしてコンテンツ名を取得する．

幅優先探索 (bfs: breadth-first search) では，DAG 上のジェネシス transaction からのホップ数の小さい transaction から順に探索する．同じホップ数の transaction を探索しても未発見の場合，一つ大きなホップ数である transaction の探索を開始し，発見するか，全探索するまで繰り返す．

深さ優先探索 (dfs: depth-first search) では，幅優先探索と同様に，ジェネシス transaction から探索を始めるが，子を持たない transaction，つまり tip に行き着くまで，深く伸びて探索する．tip に到達しても未発見の場合，最後に分岐した箇所まで戻り，未探索の transaction を探索する．

ハッシュチェイン法，二分探索木では Prefix と ID をハッシュテーブルや二分木で管理しているため，DAG 上に直接アクセスしてコンテンツ名を取得する必要があるのに対し，幅優先探索，深さ優先探索では，transaction にコンテンツ名が管理されているため，発見した時点で探索が終了する．

提案方式では，以上の四つの手法間で検索時間と，必要メモリ量を比較する．Publisher がコンテンツ名を登録する際，重複するコンテンツ名を持つ transaction が既に DAG で管理されているか確認するため台帳内を全探索し，既に存在していれば登録を拒否し，そうでなければ登録する．ここでは全 transaction における，検索時間の平均値，中央値，そして最大の検索時間の指標として上位 95%値で評価する．また，必要メモリ量では，DAG 上で直接データを管理する幅優先探索および深さ優先探索を DAG としてまとめ，DAG に加えてハッシュテーブル，二分木で管理するハッシュチェイン法，二分探索木の三つに対して比較する．

第4章　性能評価

4.1　評価条件

提案方式を計算機シミュレーションにより評価する．本研究では，IOTA シミュレータである DAGsim[11] に変更を加え，シミュレーションを行った．

また，提案方式における評価条件を以下に示す．

DAG 生成時では，URS，URW，WRW の三つの tip 選択アルゴリズムに対し，URS と URW については transaction 数 N_t を 100, 1000, 7131 の三通りで，WRW については 100, 1000 の二通りで DAG を生成する．新規コンテンツ名登録による transaction 生成はレート $\lambda_1 = 50$(/秒) の指数分布に従い発生させる．また遷移確率の式 (3.1) における α を 0.1 とする．

Consumer による要求は，総要求数を 5000 とし，発生レートを $\lambda_2 = 50$(/秒) の指数分布に従い発生させる．

またハッシュチェイン法におけるテーブルのバケット数 B を 100 とする．

4.2　コンテンツ名登録時の検索時間

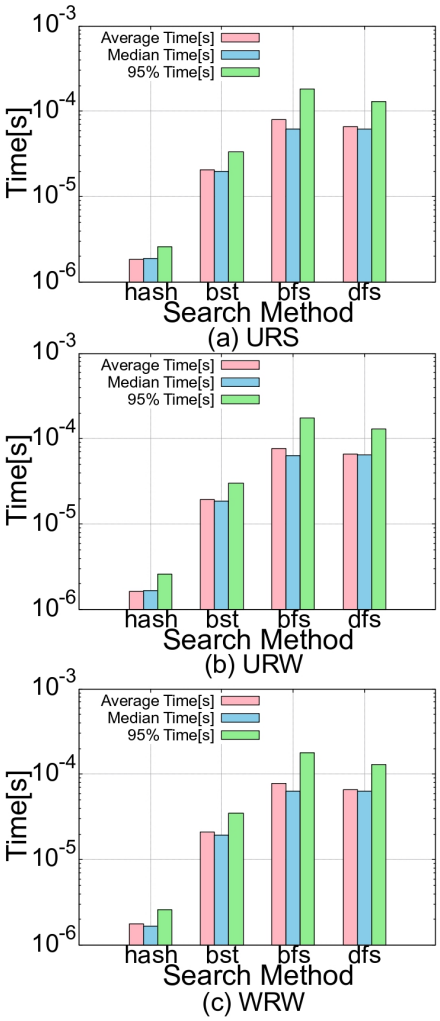

図 4.1: $N_t = 100$ におけるコンテンツ名登録時の検索時間

図 4.1 に，$N_t = 100$ における (a)URS，(b)URW，(c)WRW でのハッシュチェイン法，二分探索木，幅優先探索，深さ優先探索のコンテンツ名登録時の検索時間を示す．いずれの場合も検索時間は，ハッシュチェイン法 < 二分探索木 < 深さ優先探索 < 幅優先探索となる．ハッシュチェイン法では，ハッシュ値を算出してバケットにアクセスし，その中で管理されているリストを探索するが，コンテンツ名登録時において，最初の方はバケットで管理されているデータ数も少ないため，他の手法と比べて短い時間で全探索が可能である．二分探索木においても，全てのデータではなく，必要な部分を探索するため，検索時間も比較的短い．一方，深さ優先探索，幅優先探索では，しらみ潰しに DAG を全探索するため時間がかかる．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ないことが確認できる．ハッシュチェイン法，二分探索木では DAG とは別のテーブルを用いて探索を行うため，DAG の形状に影響を受けない．また，幅優先探索，深さ優先探索においても，少ない transaction 数では，DAG の形状に差があまりないため，

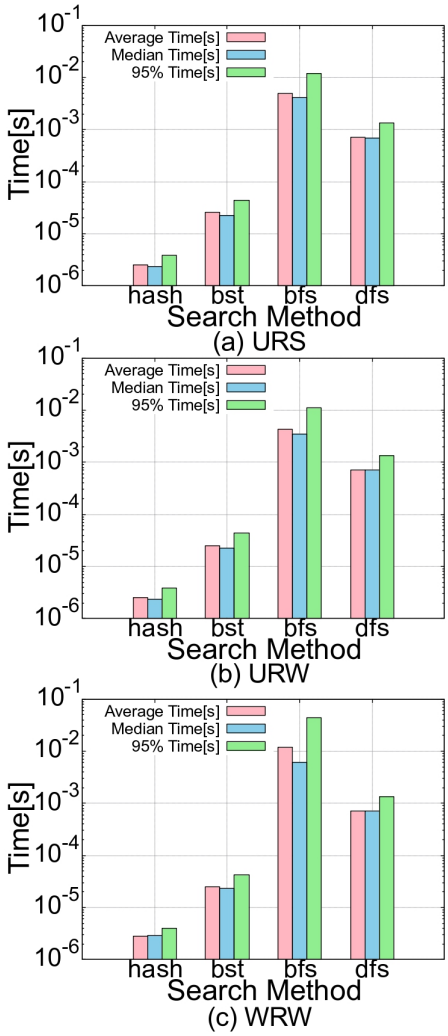

図 4.2: $N_t = 1000$ におけるコンテンツ名登録時の検索時間

結果として検索時間に差が生じにくい．

図 4.2 に，$N_t = 1000$ におけるコンテンツ名登録時の検索時間を示す．$N_t = 100$ と比べると DAG の規模が大きくなるため，いずれの tip 選択アルゴリズムでも，DAG とは別にデータを管理しているハッシュチェイン法および二分探索木と，DAG で直接管理している幅優先探索，深さ優先探索との検索時間の差が顕著になっている．図 4.2(c) の幅優先探索に着目すると，図 4.2(a)，(b) に比べて検索時間が大きいことが確認できる．これは，WRW では tip の重みが大きいものが優先して選択され，ジェネシス transaction からのホップ数が大きな transaction が多くなるため，幅優先探索では全探索に時間を費やす．深さ優先探索では異なる tip 選択アルゴリズムでも差は見られないが，この手法では，ホップ数の大きな transaction から優先して探索されるためだと推測される．なお，ハッシュチェイン法，二分探索木は検索時間が DAG の形状に依存しないため，同様にあまり差は生じない．

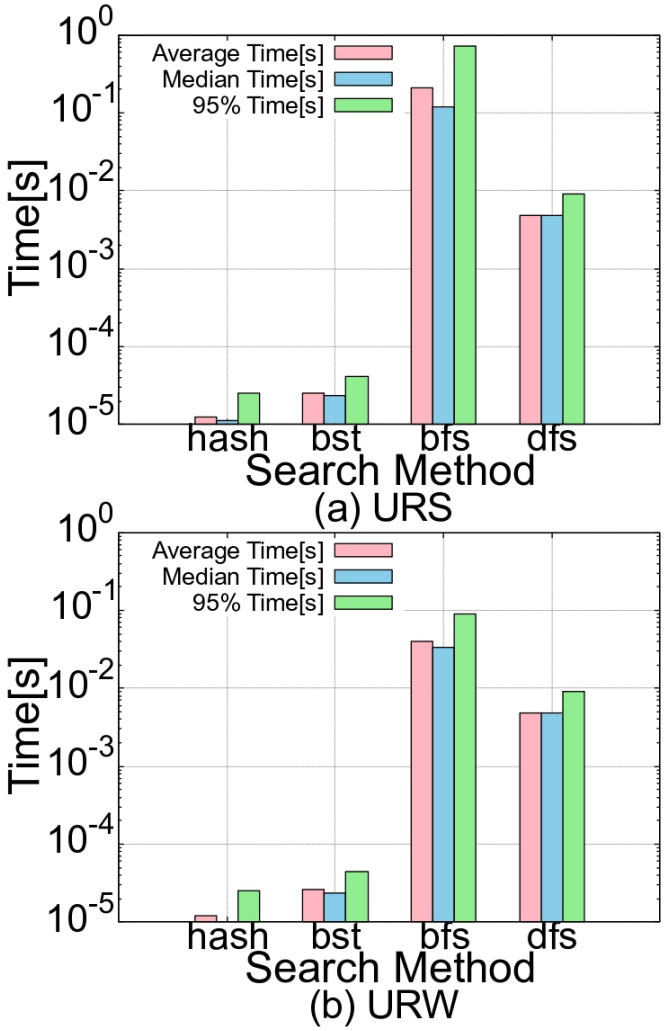

図 4.3: $N_t = 7131$ におけるコンテンツ名登録時の検索時間

図 4.3 に，$N_t = 7131$ におけるコンテンツ名登録時の検索時間をプロットする．transaction 数が増加するとハッシュチェイン法，二分探索木と幅優先探索，深さ優先探索との差はさらに大きくなることが分かる．ここで幅優先探索において，(a) の方が (b) よりも探索に時間を費やしていることが確認できる．これは，(a) では存在している tip の中からランダムに二つ選択するため，ホップ数の大きな tip でも選択されるが，(b) ではジェネシス transaction から等確率に transaction を選択するため，ホップ数の小さな transaction が DAG 上に多く存在するためだと考えられる．

4.3　名前解決時の検索時間

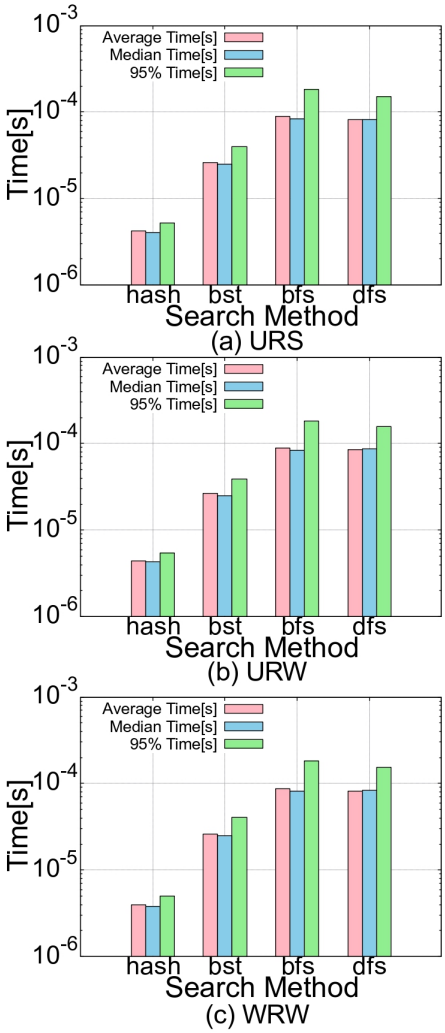

図 4.4: $N_t = 100$ における名前解決時の検索時間

図 4.4 に，$N_t = 100$ における名前解決時の検索時間を示す．比較の結果，ハッシュチェイン法＜二分探索木＜深さ優先探索＜幅優先探索となっている．こちらでもやはり transaction 数が少ないため，(a)，(b)，(c) を比較しても，tip 選択アルゴリズムによって検索時間にほとんど差が見られない．コンテンツ名登録時の図 4.1 と比較しても，検索時間にほとんど差がないことが確認できる．

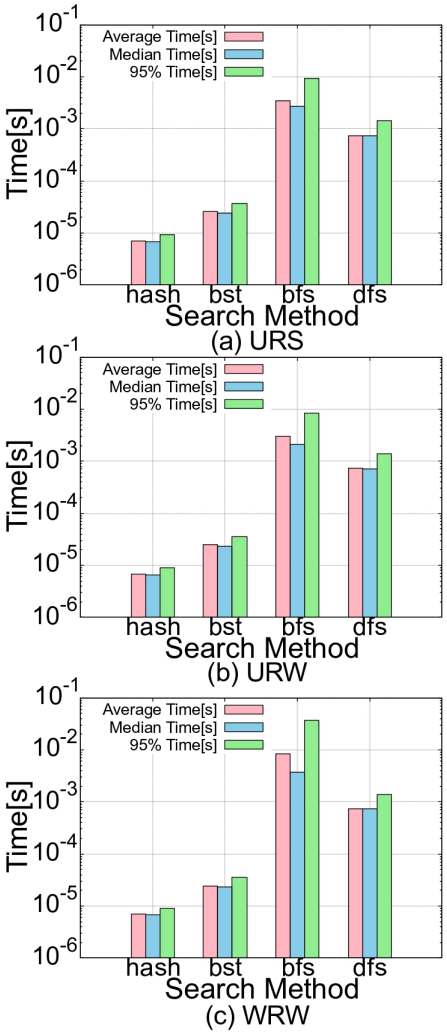

図 4.5: $N_t = 1000$ における名前解決時の検索時間

図 4.5 に，$N_t = 1000$ における名前解決時の検索時間をプロットする．transaction 数が増加するため，図 4.4 と比較して，DAG で管理する幅優先探索，深さ優先探索と，テーブルなどで管理するハッシュチェイン法，二分探索木との差が大きい．また，ジェネシス transaction からのホップ数が多い transaction が多く存在するため，図 4.5(c) の幅優先探索では (a)，(b) と比較して探索に時間を必要とする．コンテンツ名登録時の図 4.2 と比較すると，同様にあまり差は見られない．

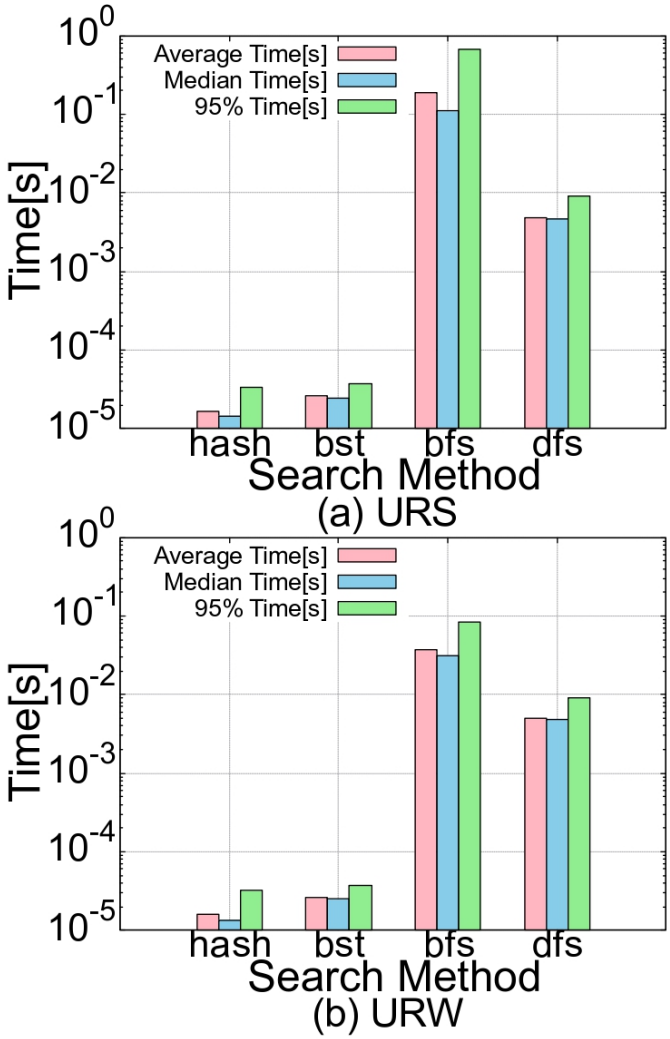

図 4.6: $N_t = 7131$ における名前解決時の検索時間

図 4.6 に，$N_t = 7131$ における名前解決時の検索時間を示す．(a)，(b) いずれの場合も，ハッシュチェイン法と二分探索木を比較すると検索時間にほとんど差がないことが確認できる．ハッシュチェイン法の平均計算量は $O(N_t/B)$ であるのに対し，二分探索木は $O(\log_2 N_t)$ と表される．つまり，transaction 数が少ないときはハッシュチェイン法の方が，多いときは二分探索木の方が計算量が少ないため，結果として検索時間の差が縮まるためだと推測される．

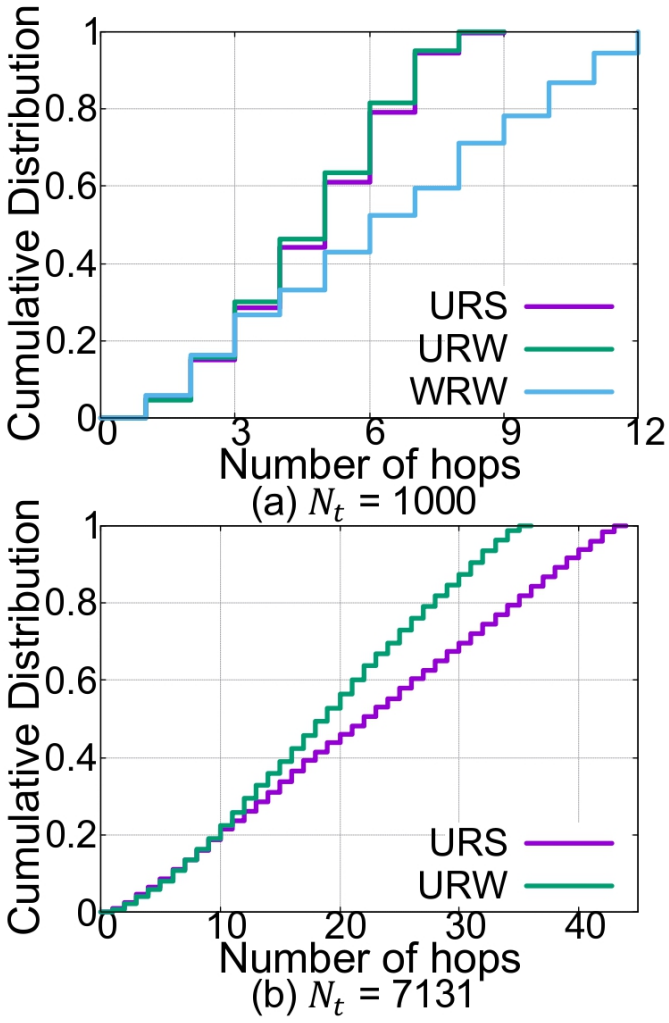

図 4.7: ホップ数

図 4.7 に，DAG における，ジェネシス transaction からの任意のホップ数を有する transaction 数に対する，累積分布をプロットする．図 4.7(a) に $N_t = 1000$ におけるホップ数の分布を示す．URS と URW は同じような分布となっており，似たような DAG の形状となっていると推測される．また，WRW ではホップ数の大きな transaction が他の二つと比べて多く，図 4.2，図 4.5 に示すように，幅優先探索ではいずれも (c) が最も時間がかかっていることが確認できる．

図 4.7(b) に $N_t = 7131$ におけるホップ数の分布を示す．URS の方が，URW よりもホップ数の多い transaction が多いことが分かる．したがって，図 4.3，図 4.6 において，いずれも (a) の方が (b) よりも幅優先探索に時間を費やすことが裏付けられる．

また，$N_t = 100$ では，ホップ数が 1 および 2 の transaction のみ存在し，異なる tip 選択アルゴリズムでも近似的に分布しているため，DAG の形状に違いはほとんどないと推測される．ゆえに図 4.1 で示すように，(a)，(b) および (c) に対し，どの探索手法でも検索時間に差が見られない．

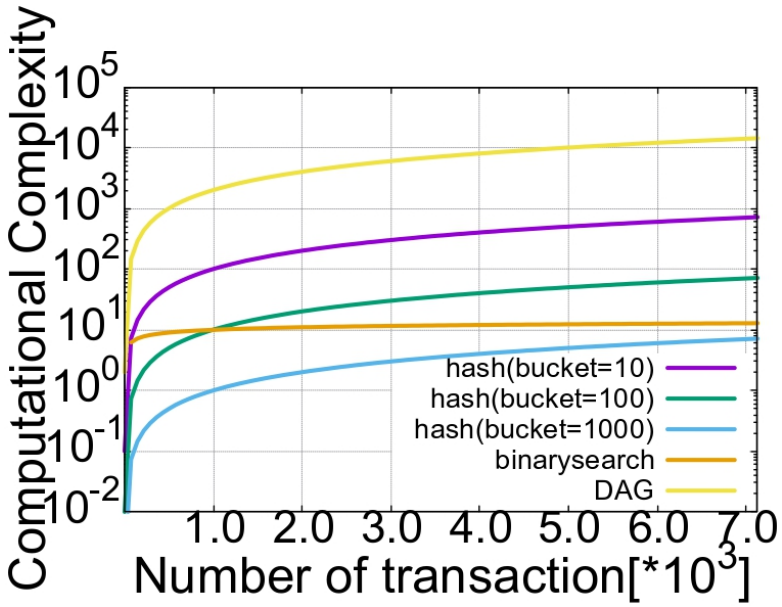

図 4.8: 平均計算量

図 4.8 に，transaction 数 N_t における平均計算量を示す．ここで，ハッシュチェイン法は $B = 10, 100, 1000$ の三通りで，幅優先探索および深さ優先探索を DAG としてプロットする．また，それぞれの探索手法の平均計算量について，ハッシュチェイン法は $O(N_t/B)$，二分探索木は $O(\log_2 N_t)$，DAG は任意の transaction が既存の二つの transaction を選択するため，エッジ数 $E = 2N_t$ とすると，$O(E)$ と表される．これらの計算量はコンテンツ名登録時，名前解決時ともに同じであると想定する．

ハッシュチェイン法では，B が増加するほど一つのバケットで管理されるデータ数が減少するため，計算量が少なくなる．本研究で行なった $B = 100$ におけるハッシュチェイン法と二分探索木を比較すると，$N_t < 1000$ ではハッシュチェイン法の方が計算量が少なく，$N_t \geq 1000$ では二分探索木の方が少なくなる．実際に，$N_t = 100$ の図 4.1 や図 4.4 において，検索時間はハッシュチェイン法の方が二分探索木よりも小さく，N_t が増加するに従って，二つの検索時間の差が縮まることが確認できる．また，DAG では，transaction 数に依存せず最も計算量が多く，シミュレーションでも同様の結果となった．

以上のことから，提案手法による該当 transaction の探索時間の結果は整合性が取れていると言える．

4.4　必要メモリ量

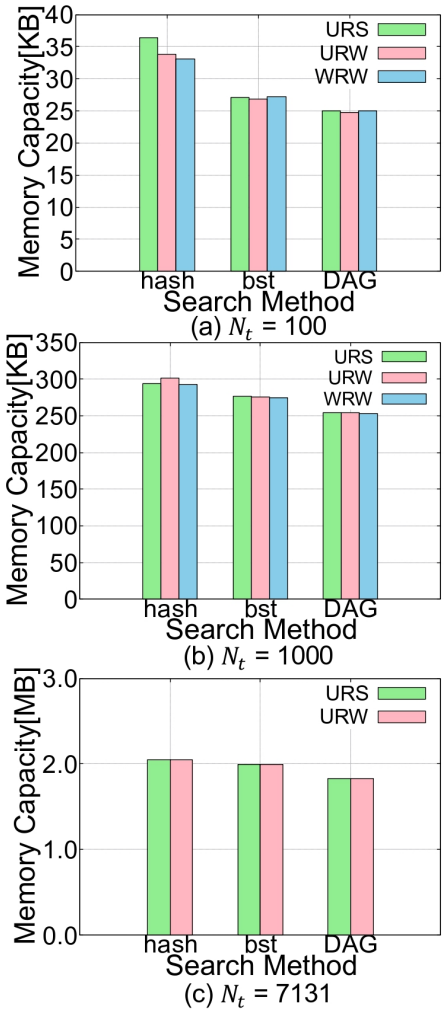

図 4.9: 必要メモリ量

図 4.9 に，探索手法ごとの必要メモリ量をプロットする．(a)，(b)，(c) のいずれの場合でも必要メモリ量は，ハッシュチェイン法 > 二分探索木 >DAG となる．ハッシュチェイン法ではハッシュテーブルで最大の容量を持つバケットの容量が全てのバケットで確保されるため，大量のメモリを必要とする．また，未使用メモリが存在するバケットもあるため，メモリ効率は悪い．二分探索木では，transaction 数分のノードの容量を必要とする．そのため，余分なメモリがなく，ハッシュチェイン法よりも必要メモリ量は抑えられる．DAG では，他のテーブルなどで管理する必要がないため，最も必要メモリ量が少ない．また，transaction 数が増加するに従ってハッシュチェイン法と二分探索木の差が縮まることが確認できる．これはハッシュチェイン法では生成データ数にかかわらず，ハッシュテーブルで固定的に大量のメモリが必要となるためである．

以上の結果から，検索時間と必要メモリ量のトレードオフの関係を確認した．

第5章　まとめ

要求されたコンテンツの名称をもとにパケットを転送し，ルータにてコンテンツをキャッシュしコンテンツ要求者 (Consumer) に配信する ICN が，次世代のネットワークとして注目を集めている． ICN では誰もが Publisher としてコンテンツをアップロード可能だが，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することでキャッシュの機能を低下させる CPA の問題が指摘されている．CPA の対策として，Consumer が公開鍵暗号を用いたディジタル署名によりコンテンツの正当性を判断可能である．しかし，公開鍵を管理する認証局の職員が攻撃者と結託し，攻撃者の公開鍵と書き換えることにより，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 は対策が困難である．そこで本論では，Publisher によるコンテンツのアップロードに際し，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し，詐称 fake 型 CPA を未然に防ぐ方式を提案した．提案方式では，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を行い，シミュレーション評価により以下のことを確認した．

- 提案方式でコンテンツ名を探索するタイミングは，Publisher によるコンテンツ名登録時と，Consumer による名前解決時の二つである．いずれの場合においても，検索時間はハッシュチェイン法が最も短く，二分探索木，深さ優先探索，幅優先探索の順に長くなる．また，transaction 数が増加するほどそれぞれの探索手法の検索時間は長くなるが，DAG の形状に影響を受ける深さ優先探索，幅優先探索では特に検索時間が長くなる．

- 必要メモリ量は，ハッシュチェイン法が最も多く，二分探索木，DAG の順に少なくなる．DAG では，他のテーブルなどで管理する必要がないため，最も必要メモリ量が少ない．一方，ハッシュチェイン法と二分探索木では DAG のメモリ量に加え，ハッシュテーブルや二分木でデータを管理するため，多くのメモリを必要とする．また，ハッシュチェイン法では，ハッシュテーブルで最大の容量を持つバケットの容量が全てのバケットで確保されるため，未使用メモリが多い．二分探索木ではデータ数分のメモリを確保するため，ハッシュチェイン法よりも必要メモリ量は抑えられる．

- 以上の結果から，検索時間の短い探索手法では多くのメモリを必要とし，検索時間の長い探索手法では必要メモリ量は抑えられる．従って，検索時間と必要メモリ量のトレードオフを確認した．

今後は Blockchain を用いてシミュレーション評価を行う予定である．

謝辞

本研究を行うに当たり，ご指導を頂いた上山教授に感謝します．また日常，有益な議論をして頂いた研究室の皆様に感謝します．

参考文献

[1] T. Nguyen, et al., ”Content Poisoning in Named Data Networking: Comprehensive Characterization of real Deployment.”, IFIP/IEEE IM 2017.

[2] W. Cui, et al.,“Feedback-Based Content Poisoning Mitigation in Named Data Networking”, IEEE ISCC 2018.

[3] P. Gasti, et al., ”DoS and DDoS in Named Data Networking”, IEEE ICCCN 2013.

[4] 工藤 多空飛, 上山 憲昭, ”ICN における Fake 型コンテンツポイズニング攻撃の影響分析”, 信学会 CQ 研究会, CQ2022-23, 2022 年 7 月.

[5] S. Popov, et al., Equilibria in the Tangle, Computers & Industrial Engineering, 136, pp.160-172, Oct. 2019.

[6] J. Park, et al., ”A Block-Free Distributed Ledger for P2P Energy Trading: Case with IOTA?”, CAiSE 2019, LNCS 11483, pp. 111-125, 2019.

[7] A. Tesei, et al., ”IOTA-VPKI: a DLT-based and Resource Efficient Vehicular Public Key Infrastructure”, IEEE VTC 2018.

[8] M. Khodaei, et al., ”SECMACE: Scalable and Robust Identity and Credential Management Infrastructure in Vehicular Communication Systems”, IEEE Transactions on Intelligent Transportation Systems 2018.

[9] G. Bu, et al., ”G-IOTA: Fair and confidence aware tangle”, IEEE INFOCOM WKSHPS 2019.

[10] S. Ghaffaripour and A. Miri, ”Parasite Chain Attack Detection in the IOTA Network”, IEEE IWCMC 2022.

[11] M. Zander. 2018. Python IOTA Tangle simulation. https://github.com/manuelzander/iota_simulation. (2022).