令和5年度　春学期　卒業研究3 (BI)
学士論文

題目　IoTのIOTAを用いたMicropayment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 26002004231
屋敷圭太

令和6年1月31日

概要

IoT では様々なプラットフォーム, デバイス所有者が相互にデータを共有, 提案することで豊富なサービスを利用することを可能にしている. IoT ではサービスの享受に伴い, 頻繁にサービス利用者とデバイス所有者, サービスプラットフォーム間で支払いが発生する. インセンティブの支払いに伴い, アプリケーションの支払いシステムとして, 少額電子決済である micropayment が注目されている. 同時に micorpayment の決済方法として分散台帳上で決済を行うことが注目されているが, micropayment による支払いは取引の頻度の観点から取引手数料が高額になるという課題が存在する. そこで本稿では, 細かい金額の粒度で支払いを行うことが出来る micropayment による取引に際し, 取引手数料が無料で高いスケーラビリティをもつ分散台帳技術の一つである IOTA 上で取引を行うことで, 支払いをまとめて行うよりも処理が軽く遅延時間が短い IoT 取引が可能な方式を提案する. 提案方式では, IOTA でコンテンツ名を検索する四つの検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, トランザクションの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

IoT(Internet of Things) は, 身の回りのあらゆるものがインターネットに繋がり, 家電やデバイスなどの日常の機器やセンサがデータを取得する. 気温, 温度, 位置情報などのセンサから得られた情報は, デバイス所有者, プラットフォームが共有する. サーバにデータが蓄積され, 収集されたデータに基づき, ユーザらは, 豊富なサービスを利用することを可能にしている. IoT ではサービスの享受に伴い, 頻繁にサービス利用者とデバイス利用者, サービスプラットフォーム間で支払いが発生する. 支払いは, 現金やクレジットカードなどの, 従来の手段を使用した支払い方法は中央集権的な金融機関や決済会社を介して処理される. 一方, 取引は決済会社, 金融機関を介して行わなければならないため, 取引の信頼性やセキュリティが第三者に依存しているという懸念が存在する. また, 通常の支払いシステムでは, 数円単位の, 金額の小さい支払いでの取引が行えず, 小規模な商品やサービスへのアクセスが困難である. そこで, アプリケーションの支払いシステムとして, 少額電子決済である micropayment が注目されている. micropayment は, 数円や数十円ほどのわずかな金額を扱う電子決済を指す. この種の支払いは, IoT デバイスなどで少量のデータを扱う場合に利用されることが期待されている. しかし, 二重支払いを防ぐための仕組みやスケーラビリティ向上のための処理の分散化が必要である. また単一の事業者がすべての支払い処理を行うとプライバシ, スケーラビリティ, セキュリティの観点で問題が発生する. これらの問題を解決するため, IoT データの micropayment の決済方法として, 分散台帳技術 (Distributed Ledger Technology: DLT) を採用した仮想通貨の使用が注目を集めている. 中でも, Blockchain が代表例である. 分散台帳技術は取引がネットワーク全体に分散されたノードによって管理され, 中央管理者が存在しないデータベースである.

1.2　研究の目的

単一の事業者が全ての支払い処理を行うことはセキュリティ, プライバシの観点で問題が発生する. そこで本稿は通常の支払いシステムよりも小さい金額で支払いが可能である micropayment による取引において, 取引手数料が無料でスケーラビリティが高い分散台帳技術の一つである IOTA [1] 上で取引を行う方式を提案する. 通常, 少額の支払いを行う場合, 手数料が相対的に大きな割合を占めることやスケーラビリティが問題となるが, micropayment に IOTA を用いることで手数料が無料でスケーラビリティの問題が解消される. 分散台帳技術としては Blockchain 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. 提案方式では, 支払いの粒度を細かくすることを想定し, transaction 数が 100, 500, 1000 の各場合における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, 必要メモリ量を比較する. そして, 検索時間と必要メモリ量のトレードオフの関係を確認する.

第2章　関連研究

2.1　IOTA

分散型台帳技術の代表例として Blockchain が挙げられるが, Blockchain では block と呼ばれるデータの集まりで構成される. 各 block には複数のトランザクションや取引データが含まれ, block ごとでデータを管理している. そのため, 一つの block に格納できる transaction 数に制限がある. また, transaction の承認にかかる手数料や, トランザクション処理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, 各 transaction ごとに処理が行われる. そのため, 他の transacction の処理の終了を待つ必要がないため, 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. では, 売り手が収集した大気質データを中央管理者の代わりに IOTA を介して, センサデータを売買する方式が提案されている [2].

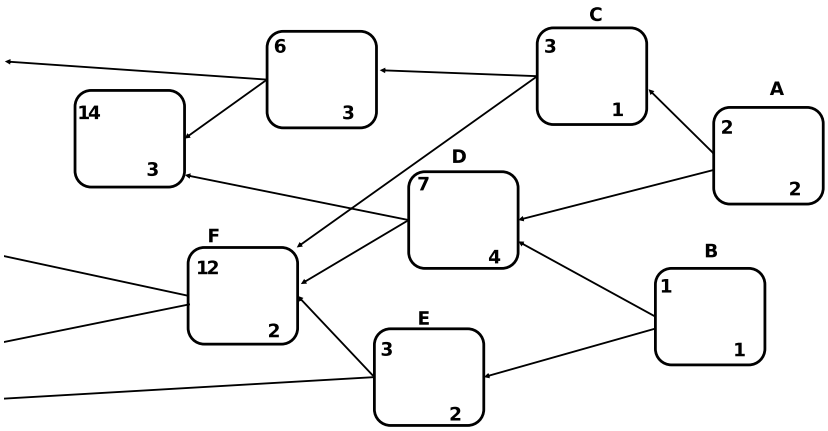


図 2.1: IOTA における DAG

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 と呼ばれる有効非巡回型グラフ (DAG: directed acyclic graph) を形成することで分散型台帳が実装される. 図 3.1 に DAG の図を示す. A, B, C, D, E, F は IOTA の transaction を表しており, それぞれ過去に Tangle に追加された 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. 累積重みは一つの transaction を直接的, 間接的に承認している transaction の重みと自身の重みを足し合わせたものが追跡重みである. 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．既存の transaction が未承認の tip を選択する方法, 手順のことを tip 選択アルゴリズムと呼び, 以下の三種類が存在する．

Uniform random selection (URS) では, Tangle 追加されている tip の中からランダムに二つ選択するアルゴリズムである．

Unweighted random walk (URW) では, Tangle の最初の transaction であるジェネシス transaction から参照されている transaction をランダムウォークによって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トランザクションから参照するが, 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 は次式で定義される．

$$P_{xy} = \frac{e^{-\alpha(H_x - H_y)}}{\sum_{z:z\to x} e^{-\alpha(H_x - H_z)}} \tag{2.1}$$

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, $\alpha(\geqq 0)$ 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なるパラサイトチェーン攻撃が挙げられる．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ほうに遷移されるように, 式の 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．

第3章　提案方式

3.1　提案方式の概要

本論文で提案するシステムは, 取引手数料無料で高スケーラビリティな取引を可能にするシステムの提案である. seller は, buyer が提供するセンサデータを購入した後, IOTA 上で micropayment を行い, 支払いを完了させる. seller は, 以下の情報を transaction に保存し, IOTA に送信する.

表 3.1: transaction 情報

用語	定義
txID	transaction の識別子
selected tips	先頭の transaction
trace	これまでに transaction が辿った経路情報
hops	ホップ数
arrival time	transaction がネットワークに到着した時刻
Prefix	IOTA で使用されるアドレス, 識別子の先頭部分
Content Name	コンテンツの名前
tip selection time	transaction が選択されるまでの時間
valid tips	正当な transaction であると証明するためのトークン
approvers	自身の transaction を承認した他の transaction 情報
buyer address	buyer の住所
payment amount	支払い金額

3.2　DAG におけるコンテンツ名探索手法

IOTA では過去に承認された transaction による取引が正常に行われたかどうか確認するため Tangle 内の transaction の検索が必要になる場合がある. そこで, 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に繋ぐことで, 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」という性質をもっている. 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, 二部探索木で発見した ID を参照して DAG 状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, 一つの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着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.

性能評価では, 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もつ transaction が既に DAG で管理されているか管理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

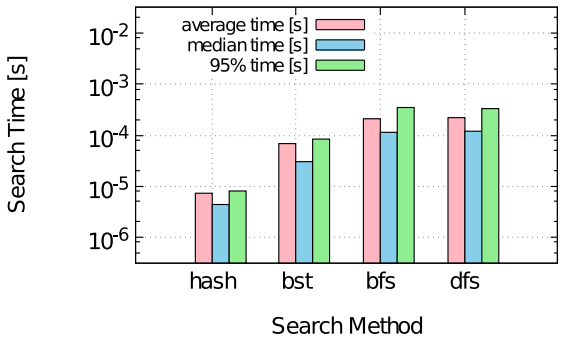
4.1　評価条件

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[3] に変更を加え, シミュレーションを行った.

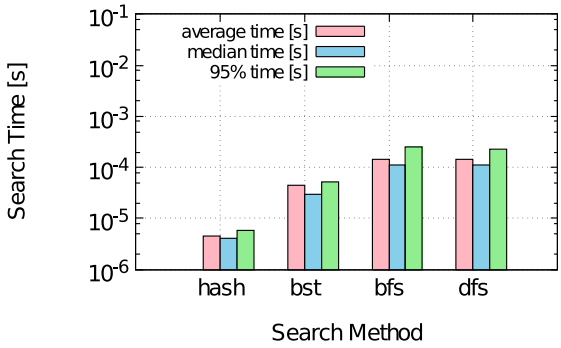
また, 提案方式における評価条件を以下に示す.

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, transaction 数 N_t を 100, 500, 1000 の三通りで, DAG を生成する．一秒当たりの transaction 生成数を 50(/秒) の指数分布に従い発生させる．また遷移確率の式における α を 0.1 とする．

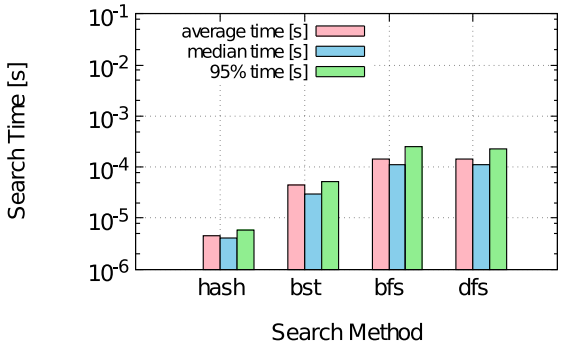
4.2　該当 transaction の検索時間



(a) URS



(b) URW

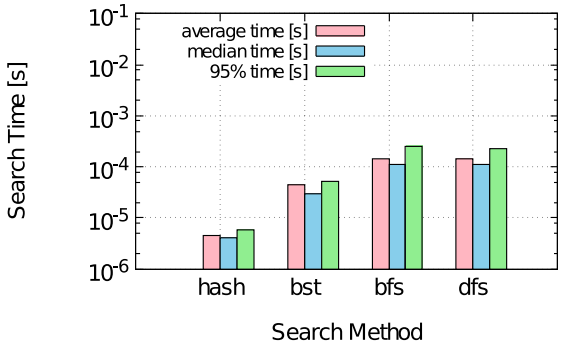


(c) WRW

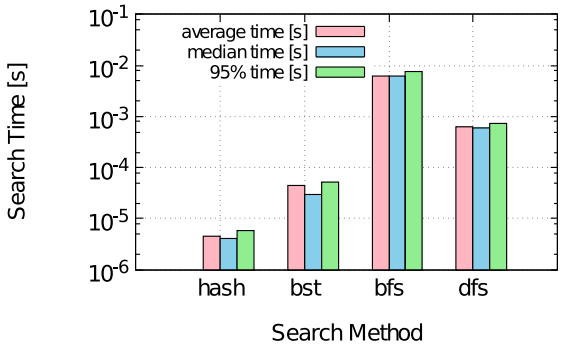
図 4.1: $N_t = 100$ における検索時間

図 4.1 に, transaction 数 $N_t = 100$ 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データ数も少ないため, ほか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

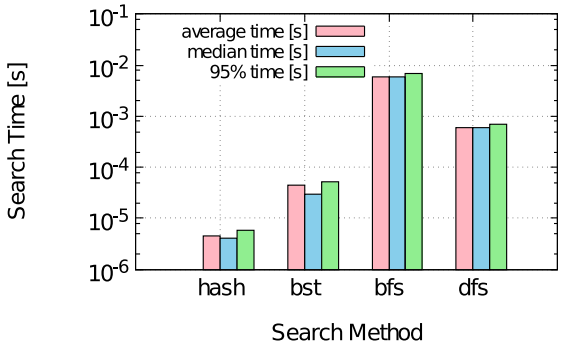
木においても, 全てのデータでなく, 必要な部分を捜索するため, 検索時間も比較的短い. 一方, 深さ優先探索, 幅優先探索では, しらみつぶしに DAG を全探索するため時間がかかる. tip 選択アルゴリズムごとに比較すると, どの手法も検索時間の差はほとんどないことが確認できる. ハッシュチェイン法, 二分探索木では DAG とは別のテーブルを用いて探索を行うため, DAG の形状に影響を受けない. また, 幅優先探索, 深さ優先探索においても, 少ない trasaction 数では, DAG の形状に差があまりないため, 結果として検索時間に差が生じにくい.



(a) URS



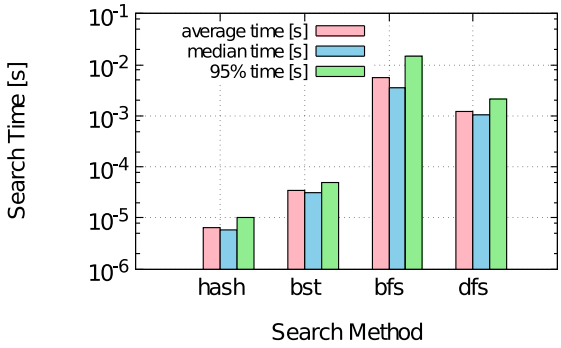
(b) URW



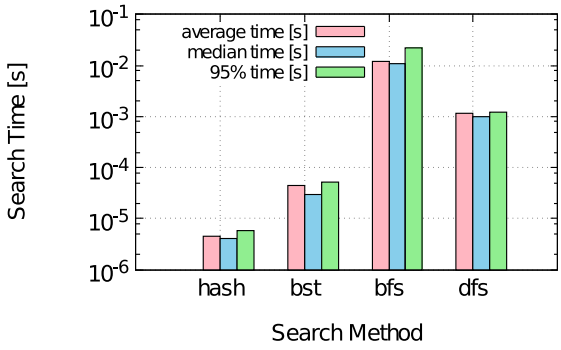
(c) WRW

図 4.2: $N_t = 500$ における検索時間

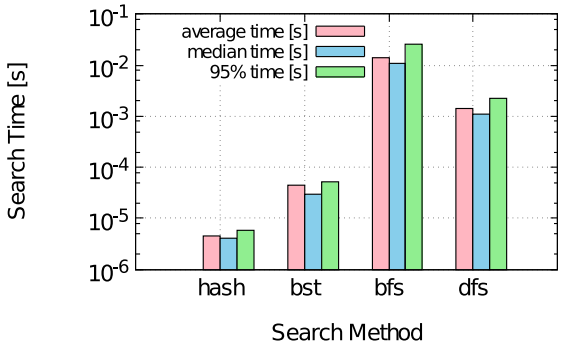
図 4.2 に, $N_t = 500$ における transaction の検索時間を示す. (b), (c) に着目すると, 幅優先探索, 深さ優先探索の検索時間が増加している. これは, $N_t = 100$ と比べて DAG の規模が大きくなったため, DAG の形状に依存する二つの探索手法の検索時間が増加したと推測される. ハッシュチェイン法, 二分探索木の探索時間に大きな差は見られない.



(a) URS



(b) URW

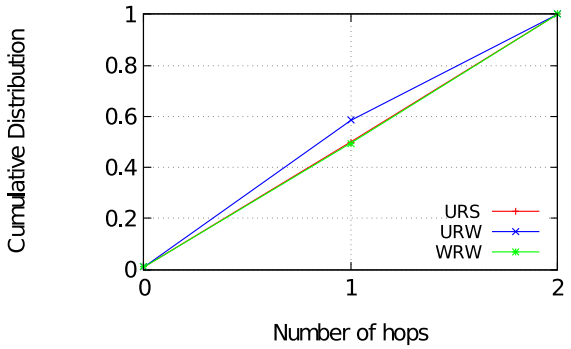


(c) WRW

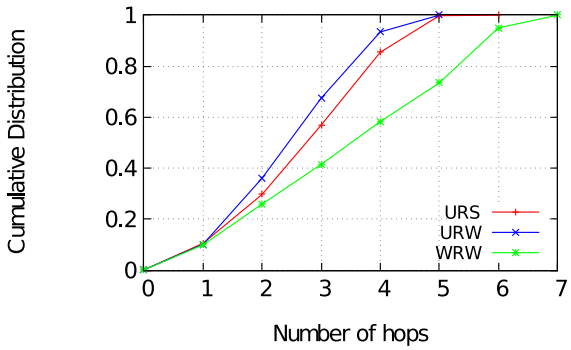
図 4.3: $N_t = 1000$ における検索時間

図 4.3 に $N_t = 1000$ における transaction の検索時間を示す. $N_t = 100$ と比較すると 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 上でデータを直接管理して

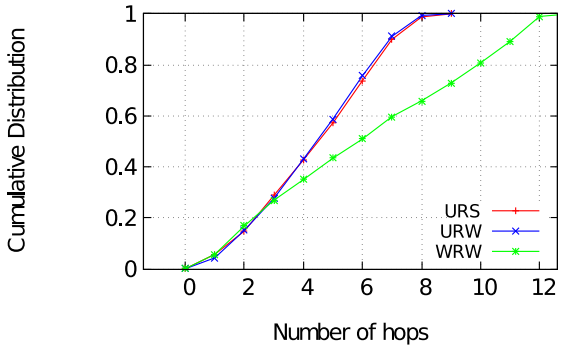
いる幅優先探索, 深さ優先探索との検索時間の差が顕著になっている. $N_t = 100$ $N_t = 500$ に比べハッシュチェイン法, 二分探索木の検索時間にあまり差は生じていない. これは, これら手法は検索時間が DAG の形状に依存しないためであると推測される. 図 4.2(c) の幅優先探索に着目すると, (a), (b) に比べ, 検索時間が大きいことが確認できる. これは, WRW では 未承認 transaction である tip が持っている重みが大きいものが優先して選択され, ジェネシス transaction からのホップ数が大きな transaction が多くなるため, 幅優先探索は他二つの tip 選択アルゴリズムよりも全探索に時間がかかる. 深さ優先探索に選択アルゴリズム間で差はない. これはこの手法においては, ホップ数の大きな transaction から優先して探索されるためであると推測される.



(a) $N_t = 100$



(b) $N_t = 500$



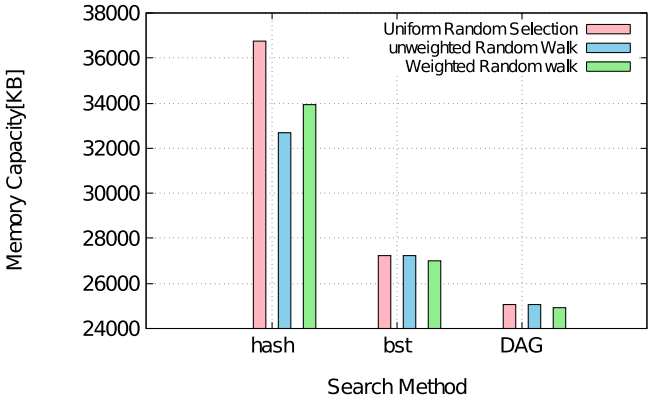
(c) $N_t = 1000$

図 4.4: ホップ数

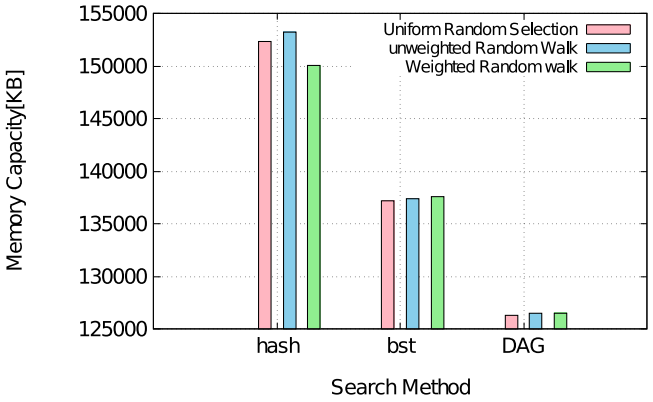
図 4.4 に, DAG における, ジェネシス transaction 数に対する累積分布をプロットする. 図 4.4(a) に $N_t = 100$ におけるホップ数の分布を示す. 3 つの tip 選択アルゴリズム間で差異が小さく, ホップ数が 1, 2 の transaction のみ存在していることから, $N_t = 100$ においては 3 つのチップ選択アルゴリズムで DAG の形状に違いはほとんどないことが推定される. よって, $N_t = 100$ ではどの tip 選択アルゴリズムでも検索時間に差が見られない. 図 4.4(b) に $N_t = 500$ におけるホップ数の分布を示す. URS に比べ URW のホップ数 1 から 4 の transaction 数が多いが, そこまで大きな違いはないため, DAG の形状に大きな

違いはないことが推測される．WRW ではホップ数の大きな transaction が URS, URW と比べて多い．図 4.4(c) に $N_t = 1000$ におけるホップ数の分布を示す．URS と URW はほぼ同じグラフの形になっているため, 似たような DAG の形状となっていると推測される．また, $N_t = 500$ と同様に WRW ではホップ数の大きな transaction が他の二つの tip 選択アルゴリズムと比べて多い．

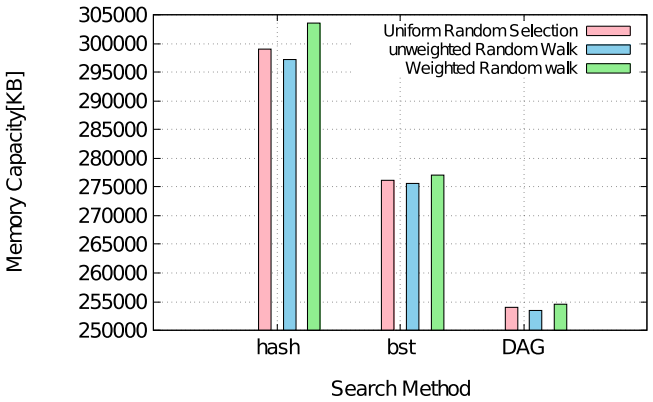
4.3　必要メモリ量



(a) $T = 100$



(b) $T = 500$



(c) $T = 1000$

図 4.5: 必要メモリ量

図 4.5 に, 各探索手法, transaction 数ごとの必要メモリ量をプロットする. いずれの transaction 数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. 二部探索木では, transaction 数分のノードの容量を必要とし, 余分なメモリは存在しないため, ハッシュチェイン法よりも必要メモリ量は抑えられる. DAG では, ほかのテーブルなどで管理する必要がないため, 最も必要メモリ用が少ない. また, ハッシュチェイン法では生成するデータ数に関わらず, 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, 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. 以上の結果より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

本論文では, micropayment での支払い処理に際し, 取引手数料が無料で, 高スケーラビリティな分散台帳技術の一つである IOTA 上で支払い処理を行い, 高頻度の取引においても取引手数料が無料である方式を提案した. 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.

- transaction の検索時間はいずれの transaction 数, tip 選択アルゴリズムにおいても, 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, 幅優先探索の順に増加する. 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増加するが, 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, DAG が最も少なく, 二分探索木, ハッシュチェイン法の順に増加する. DAG では, 別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という結果になった. それに比べ, 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,
- 以上の結果より, ハッシュチェイン法と二分探索木のような, 検索時間の短い探索手法では多くのメモリを必要とし, 深さ優先探索と幅優先探索のような検索時間の長い検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.

今後は実機による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] S. Popov, et al., Equilibria in the Tangle, Computers Industrial Engineering, 136, pp.160-172, Oct. 2019..

[2] R. Nakada, Z. Li, T. Pei, K. Nguyen and H. Sekiya, "An IOTA-Based Micropayment System for Air Quality Monitoring Application," 2021 IEEE 94th Vehicular Technology Conference (VTC2021-Fall), Norman, OK, USA, 2021, pp. 1-6, doi: 10.1109/VTC2021-Fall52928.2021.9625358.

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

学会発表リスト

・屋敷圭太, 上山憲昭,”IoT の IOTA を用いた Micropayment 方式”, 電子情報通信学会 2023 年総合大会, 広島,2023 年 3 月