千葉工業大学
修士学位論文

時空間永続証明のための
ブロックチェーン履歴交差法の検討

所属専攻：電気電子情報工学専攻
学籍番号・氏名：1972042　柳原　貴明

指導教員：藤原　明広 准教授

提出日 令和 3 年 3 月 dd 日

概要

最後に書く

目次

1 序論

近年，クラウドコンピューティングおけるデータサーバーの仮想化を代表されるようなストレージやネットワーク，アプリケーションなどの様々な機能が仮想化技術によって提供されている．利用者は，メンテナンスフリーや維持費の安さなど利点は多くあり普及が加速しているがデータサーバーを管理する大きな力を持った企業・組織に依存することになる．

Bitcoin を代表とする (パブリックな) ブロックチェーンは，世界中に散在するコアノード同士で，取引データやブロックを転送するため頻繁に共有し合っている．地理的に近いコアノード同士では通信遅延時間の影響は少ないが，遠距離のノードにブロックを転送する場合，数十秒〜数分の通信遅延が起きる．

現状では，Proof of Work（ PoW ）によるブロック生成時間を十分遅くする必要があるこの理由によりビットコインではブロック生成時間が 10 分になるように制御されている．従って遅延時間により，単位時間に処理可能な取引量が少なくなる．これは「スケーラビリティ問題」として知られている．この問題を解決する方法の一つとして，地理的に近いコアノード同士でドメインを自律的に形成し，ドメイン毎に独自のブロックチェーンを管理する方法が提案されている [2][3]．しかし，各ドメインに存在するコアノードの数が減少するため，ブロックチェーンの非中央集権性と改ざん耐性が低下する問題があった．本章では，本研究の背景として現状の動向と先行研究の問題点を挙げ，本研究目的を述べる．

1.1 背景

Bitcoin: A Peer-to-Peer Electronic Cash System の誕生

近年，クラウドコンピューティングおけるデータサーバーの仮想化を代表されるようなストレージやネットワーク，アプリケーションなどの様々な機能が仮想化技術によって提供されている．利用者は，メンテナンスフリーや維持費の安さなど利点は多くあり普及が加速しているがデータサーバーを管理する大きな力を持った企業・組織に依存することになる．金融界における Bitcoin 誕生以前は，インターネットバンキング等の電子取引は，ほぼ例外なく，信用することのできる第三者機関に金融機関に依存しているのが現状であった．Bitcoin は，信用ではなく暗号化された証明に基づく電子取引システムであり，これにより希望する二者が信用できる第三者機関を介さずに直接取引ができる．従って，

金融界の中央集権性を否定している．非中央集権性を可能にする技術がブロックチェーンである．ブロックチェーンは，金融界だけでなく様々な業界での応用が期待されている．

Bitcoin ブロックチェーンのスケーラビリティ問題

Bitcoin を代表とする (パブリックな) ブロックチェーンは，世界中に散在するコアノード同士で，取引データやブロックを転送するため頻繁に共有し合っている．図 1 に 2020 年 11 月 17 日現在の全世界の Bitcoin コアノードの分布を示す．

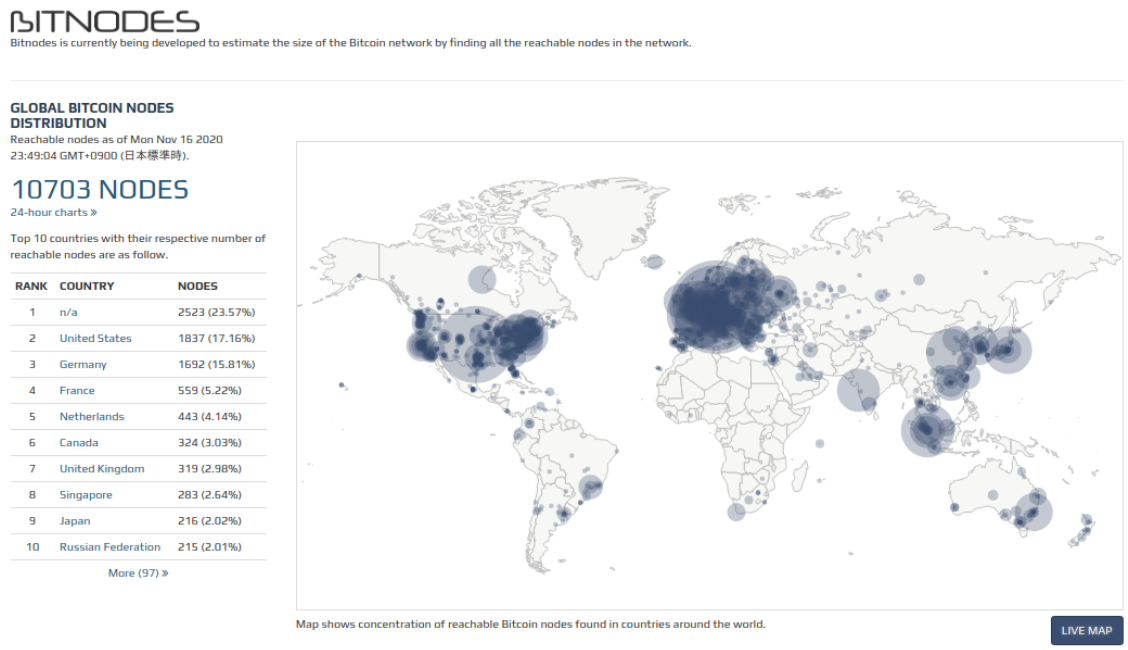

図 1　2020 年 11 月 17 日現在の全世界の Bitcoin コアノード分布

地理的に近いコアノード同士では通信遅延時間の影響は少ないが，遠距離のノードにブロックを転送する場合，数十秒〜数分の通信遅延が起きることが知られている．この通信遅延により，Proof of Work（ PoW ）によるブロック生成時間を十分遅くする必要がある従って，ビットコインではブロック生成時間が 10 分になるように制御されている．この遅延時間により，単位時間に処理可能な取引量が少なくなる．これは「スケーラビリティ問題」として知られている．

スケーラビリティの問題へのアプローチ

ブロックチェーンのスケーラビリティ問題に対するほとんどの解決策は，コンセンサスプロトコルの改善，ブロックサイズの縮小，ネットワークスケーリングのいずれかになる．ブロックサイズのサイズの縮小は，Off-chain の考え方であり，情報量を必要最低限に削減し，通信遅延等のスケーラビリティ問題への解決を目指す提案が行われているが，スケーラビリティ問題の本質的な改善にはならない．本研究では，従来のブロックチェーンの情報量を削減しない（On-chain）方法でのスケーラビリティ問題を改善を行う．

1.2 本研究の目的

先行研究では，ドメイン分割を行い通信距離による通信遅延を削減し，地理的に近いコアノード同士でドメインを形成し，ドメイン毎に独自のブロックチェーンを管理する方法が提案されている．[2][3]．しかし，単にドメイン分割を行うと各ドメインのブロックチェーンに関わるコアノードの数が減少する．これにより，ブロックチェーンの特性である非中央集権性と改ざん耐性が低下する問題がある．そこで本研究では，ブロックチェーンの状態を定期的に共有し，各ドメインのブロックチェーンに互いの署名 (ヒステリシス署名) を書き込み合う履歴交差法を提案し，先行研究で挙げれる非中央集権性低下と改ざん耐性の低下してしまう問題を改善することを目的としている．

2 関連研究

2.1 BBc-1

BBc-1[5] は，プライベートな取引データを履歴交差により改ざん耐性を持たせて保存する (ブロックチェーンではない) 分散台帳技術である．ノードの実装も行われており，その有効性が実験的にも確認されている．BBc-1 では，取引データにヒステリシス署名を追加する．ヒステリシス署名を取引データに付けて，ドメイン間でデータを共有することで改ざん耐性を高めている．また，PoW による改ざん耐性の向上を行わない為，エネルギーコストは低く済む特徴がある．しかし，改ざん耐性がどの程度向上したかを理論的に評価しにくいことも知られている．

2.2 bloXroute

bloXroute [6] では，P2P ネットワーク上で取引データやブロックの共有速度を早める為に，構造化されたネットワークを (Layer0 と呼んでいる) 上位ネットワークに構築する提案を行っている．Plasma や Sharding のようなプロトコルがレイヤー 2 のソリューションとみなされるなら，bloxroute はレイヤー 0 のスケーラビリティソリューションとして分類される．これらのレイヤ 2 スケーラビリティ・プロトコルの多くは，現在の世代のブロックチェーン技術で広く採用されるようになる可能性が高いが，空間の細分化にも貢献している．bloxRoute はそれらの従来のモデルを無視し，あらゆるブロックチェーンの下で動作するソリューションを提案している．

bloxRoute は, レイヤー 0 のプロトコルで, ブロックチェーンの全体的なアーキテクチャに影響を与えることなく，オンチェーン全体で数桁の増加を可能にします．bloXroute BDN は，ブロックチェーンの下で分散型ルーターとして機能することで，高いレベルのスケーラビリティを実現している．

bloXroute は，スケーラビリティ問題に対するアプローチは，既存のブロックチェーンの基礎となるプロトコルを崩すことなく，ブロックチェーンネットワークにおけるスケーラビリティの問題を解決する独自のアプローチを提案している．bloXroute は，ブロックチェーンに関しては対立することが多い 3 つの原則，すなわちスケーラビリティ，中立性，適応性を兼ね備えている．このプラットフォームは，あらゆるブロックチェーンに組み込むことができる効果的なブロードキャスト・プリミティブを実装することで，スケーラビリティを提供している．暗号化されたブロックをサポートすることで中立性を実現

し，ピアリレーでブロックの出所を不明瞭にすることで中立性を実現している．最後に，ユーザがゲートウェイを介してネットワークを系統的に直接かつ積極的に調査できるようにすることで，監査可能性を実現している．

2.3 PoWaP（Poof of Work at Proximity for a crowdesnsing system）

PoWaP とは地理的に近いノード同士が空間領域（ドメイン）を構成して蜜になるネットワークを構成することで，ドメイン毎に独自のブロックチェーンを管理するネットワークシステム [?] に概要図を示す．ドメインに分割して制御することで，ブロックの生成する時間が短縮可能である．また，利用者は近くのノードに近距離無線通信でアクセスして取引等の情報を送ることで，ドメイン（地域）に特化した情報をブロックチェーンに保持させることも可能になる．

2.4 Atomic Swap

Atomic Swap は異なるチェーン間で相手を信用することなく，アトミックにコインを交換するのに使用されるプロトコルである．一般的なプロトコルとしては HTLC が利用される．

アリスが Bitcoin を持ち，ボブが Litecoin を持っている状態でアリスとボブはそれぞれ Bitcoin と Litecoin を交換したい場合，以下のような手順で相手を信用することなく各チェーンのコインをアトミックに交換できる．

1. アリスはシークレット A をランダムに選択し，そのハッシュ値 H(A) をボブに伝える．
2. アリスは自分の Bitcoin を以下のいずれかの条件でアンロック可能なスクリプト宛に送金するトランザクションを作成し Bitcoin ネットワークにブロードキャストする．
   - H(A) のプリイメージを提供すればボブの鍵でアンロック可能.
   - 2 日後アリスの鍵でアンロック可能.
3. ボブは自分の Litecoin を以下のいずれかの条件でアンロック可能なスクリプト宛に送金するトランザクションを作成し Litecoin ネットワークにブロードキャストする．
   - H(A) のプリイメージを提供すればアリスの鍵でアンロック可能.
   - 1 日後ボブの鍵でアンロック可能.

4. アリスはシークレット A を使って 3 のボブの Litecoin を入手するトランザクションを作成し Litecoin ネットワークにブロードキャストする．
5. ボブは Litecoin のブロックチェーン上で 4 のトランザクションを確認することでシークレット A の値がわかる．
6. ボブはシークレット A の値を使って 2 のアリスの Bitcoin を入手するトランザクションを作成し Bitcoin ネットワークにブロードキャストする．

同様に本研究におけるドメイン間で分割を行った異なるブロックチェーン間の取引データも，Atomic Swap で用いることで情報のやり取りができる．

2.5 Raft Consensus Algorithm

[11] 複数のノード間で合意を得る必要がある．分散システムで合意するを得る手段に Raft がある Raft は Paxos ど同等の性能を持ちながら Paxos よりも格段の理解のしやすさを提供する分散合意プロトコルである．Raft を実行するノードには, LEADER, CANDIDATE，FOLLOWER という 3 種類の状態がる．LEADER はクラスタ内に 1 台しか存在しない．クライアントは，LEADER ノードとインタラクションする．クライアントは LEADER にアクセスしてきた場合は LEADER にリダイレクトされる．FOLLOWER のノードは LEADER ノードに従う．LEADER ノードはクライアントノードから更新要求を受信すると，ログレコードを作成する．そして LEADER ノードはログレコードをストレージに保存した後，ログレコードをログレコードを FOLLOWER ノードへ転送する．FOLLOWER ノードは受信したログレコードをストレージに保存すると，ACK を LEADER へ送信する LEADER を含め，Raft クラスタの過半数が命令を受け取ったことを LEADER が確認した後，クライアントに COMMIT を通知する．最終的には，すべてのノードが同じ命令を同じ順番で実行するため，すべてのノードでデータが複製される．即ち Raft は複製化状態機械に基づいて設計されている．一般に利活用されている複製化状態機械に例として，Chubby，ZooKeeper などが挙げられる．LEADER ノードが故障したり，LEADER ノードの故障したり，LEADER ノードのネットワークコネクションが切断されると，Raft クラスタ内に LEADER が存在しなくなる．このような場合，FOLLOWER ノードは状態を CANDIDATE へと変更する．クラスタ内のノードは各 CANDIDATE への投票を行い，新たな LEADER を選出する．この選出にはクラスタを構成するノードの数の過半数の選任が必要である．過半数を得たノードが存在しなければ，LAEDER 選出作業を再実行する．

3　関連技術動向

ソフトウェアシステムの構造（アーキテクチャ）には大きく「集中システム」と「分散システム」の２つがある．図のように集中システムは１つの中心があり，そこに他のコンピュータが接続する形になっており，

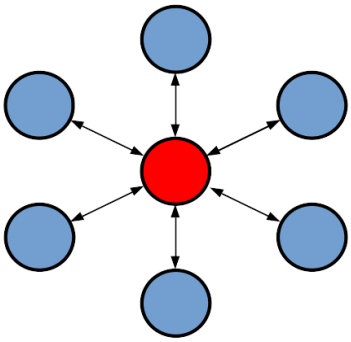

図２　集中システム

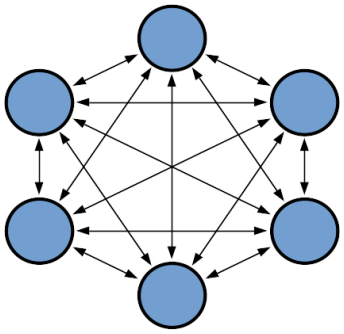

図３　分散システム

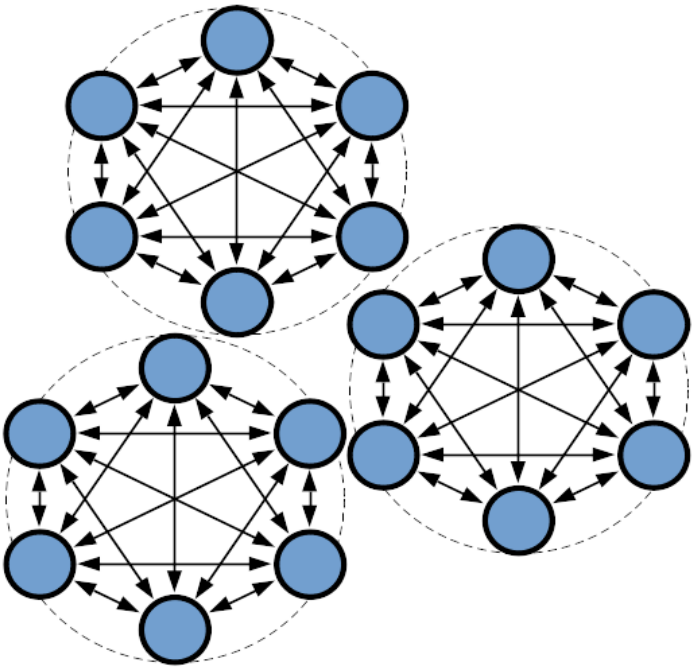

図４　ドメイン分割

図のように，集中システムは１つの中心があり，そこにほかのコンピューターが接続す

る形になっており，分散システムは複数のコンピューター同士が相互に接続し合って形成される．システムによっては，分散システムと集中システムを組み合わせることもあり，状況に応じて適切な構造を取るように工夫されてる．集中システムでは，中心のコンピューターが動かなくなるとシステム全体が停止（システムダウン）してしまう「単一障害点」と呼ばれるマシンが存在したり維持管理コストがかかってしまったりする欠点があるものの，システムの柔軟性や一貫性を維持するのが比較的，容易である長所がある．また何よりもシステムにおけるデータ完全性を維持できるメリットがある．分散システムは，システムにおける処理の多くのコンピューターで分担するためコストを削減することがで，単一障害点がないことでシステムダウンする可能性がとても低くなる利点がある．加えて，計算能力を高めていくことも可能になる．ネットワークに接続している多くのコンピューターの計算能力を利用できるため全体の計算能力を徐々に大きくすることや，今では多くの人々が高性能の PC やスマートフォンなどを持つようになってるため，それらが接続し合って大きな計算能力を実現するようになった．また低コストかつダウンしないシステム分散システムは，複数のコンピューター同士が接続し合って形成される．システムによっては，分散システムと集中システムを組み合わせることもあり，状況に応じて適切な構造を採るように工夫されている．

3.1 Bitcoin

Bitcoin はデジタルマネーエコシステムの基礎となるコンセプトと技術の集合体である．Bitcoin ネットワークの参加者間で，価値の保有と転送が Bitcoin という名の通貨単位で行われる．ユーザー間の通信は主に Bitcoin プロトコルに基づいて，インターネットを通じて行われる．Bitcoin のプロトコルスタックはオープンソースとして利用可能であり，ノートパソコンからスマートフォンまで，様々なデバイス上で動作する．

ユーザーはネットワークを通じて Bitcoin をやり取りすることで，従来の通貨で行うほぼ全てのこと，つまり物品の売買から個人や組織への送金，融資まで行うことが可能になる．Bitcoin 自体も売買が可能であり，専門の両替機関で他の通貨とも両替することも可能です．Bitcoin は高速かつ安全であり，国境を越えて取引が可能であることから，ある意味でインターネットに使うための最適な通貨とも言えるかもしれない．伝統的な通貨と異なり Bitcoin は完全に仮想的なものです．物理的なコインは存在せず，またデジタルコイン自体が存在するわけでもありません．コインは「送信者から受信者へある一定量の額面を移動させる」という取引 (トランザクション) の中で示されるものである．

Bitcoin のユーザーはトランザクションの所有権を証明する鍵を所有し，そのトランザ

クションの中に記載された額面を使用したり新しい所有者に送金することができる．この鍵はそれぞれの利用者の PC 上の電子財布 (ウォレット) に保持される．この鍵の保有が Bitcoin を使用する唯一の条件であり，そのコントロールは各ユーザーにゆだねられている．

Bitcoin は分散された peer-to-peer のシステムであり，何らかの”中央”サーバや管理者が存在するものではない．Bitcoin は取引の過程で行われる”マイニング”と呼ばれる数学的な解を見つけ出す競争により新たに生み出される．どの (フルプロトコルスタックを動作させている)Bitcoin ネットワーク参加者も，自身のコンピューターリソースを用いて取引の記録処理と検証処理を行うことでマイナーとなることができる．平均して 10 分に 1 回の頻度で誰かが数学的な解を見つけることで取引が検証され，その解の発見者には Bitcoin が新たに与えられいる．従って Bitcoin のマイニングとは，中央銀行が行う必要があった通貨の発行と決済の機能を，世界的な競争で代替したものである．Bitcoin プロトコルにはマイニングの機能を規定するアルゴリズムが組み込まれている．マイナーが行わなければならないタスクの難易度は，マイナーの数 (または CPU の数) が変動しても平均的に 10 分に 1 回ほど解が見つかるように自動的に調整されいる．またプロトコルでは Bitcoin が新たに発行される頻度は 4 年毎に半減されるように規定されており，また Bitcoin の総発行量は 2100 万 Bitcoin を超えないように規定されている．その結果，Bitcoin の流通数は容易に予想可能で 2140 年に 2100 万に到達するカーブを描くことになる．Bitcoin はプロトコル名でもありネットワーク名でもあり，さらには，分散コンピューティングのイノベーションの名称でもある．

3.2 ブロックチェーン

ブロックチェーンは 2008 年に「Satoshi Nakamoto」という仮名の人物ないし集団が発表がた Bitcoin の基幹技術として生まれ，現在は主に仮想通貨としての活用を目指す技術として注目を浴びている．

ブロックチェーンは 2008 年，匿名の開発者サトシ・ナカモトにより，デジタル通貨システム「Bitcoin（ビットコイン)」を実現するための分散タイムスタンプとして提案された．ビットコインは，管理者のいない環境下で電子的に表現された通貨の制御権の移転，すなわち送金を妨害されないことを目標に設計されている．送金の取引は入金 (入力) と出金 (出力) の間の関係を記述するが，ビットコインでは，未使用の取引出力がコインであるとするいわゆる

UTXO(Unspent TX Output) 構造 (図 -1) を用いて電子コインのデジタルデータ形式

を定義している．この構造は，適切な秘密鍵を用いて取引にディジタル署名できる主体だけが送金できることを保証するが，署名の検証に必要な情報 (公開鍵) をデータ構造の中に埋め込み，公開鍵のメッセージダイジェスト (ハッシュ値) をコインの送金の宛先とすることで，まったく関係のない第三者でも公開鍵の正当性を確認でき，取引の形式的な正しさを検証可能にした点である．

仮想通貨のひとつであるビットコインはブロックチェーンによって通貨の取引を管理するが，そこで利用されている暗号学的ハッシュ関数は，$SHA-256$ と $RIPEMD160$ である．ブロックチェーンとは，通貨の取引履歴のように改ざんされないよう管理されるべきデータを扱うための，分散型（$P2P$ 型）の台帳を実現するための技術である．

ビットコインにおけるブロック構造

一定数のトランザクションを格納したものをブロックを呼ぶ．ブロックには，多数のトランザクションとあとで説明する「ナンス」(Nonce) と呼ばれる特別な値，そして直前のブロックのハッシュ値を持っている．「ブロック」に含まれた取引のみを「正しい取引」と認めることにする．そしてネットワーク全体で「唯一のブロックの鎖」を持つようにする．これによって「一貫した取引データを全体で共有できる」というのがブロックチェーンの考えである．

ビットコインブロックチェーンにおけるブロック構造は以下図のように構成されている．

3.2.1 transaction（トランザクション）

トランザクション

3.2.2 Nonce（ナンス）

3.2.3 hash 値（ハッシュ値）

3.3 P2P ネットワーク [peer to peer NetWork]

P2P ネットワークは，Peer（ピア）と呼ばれる対等な立場のコンピューター同士が相互にデータを融通し合って形成されるネットワークです．P2P ネットワークの性質として，

1.「システムがダウンしない」
2.「ネットワークの分断耐性が強い」
3.「データの一貫性を維持することが難しい」

という点が挙げられる．

Proof of Work (PoW)

不特定多数のコンピューターによる演算を行いブロックチェーン全体の整合性を保つためのアルゴリズム具体的には，取引データを要約したデータやタイムスタンプなどのデータをひとまとめにしそのデータにナンス（Nonce）を加えて，ハッシュ関数にかけてハッシュ値を求める．このハッシュ値が一定値（目標値）よりも小さな値になるまでナンスを変えて計算する．このことを「マイニング」と言い，マイニングを行う主体をマイナーと呼ぶ．

ナンス以外のデータは固定されているので，ナンスの値を少しずつ変えるだけになる．しかし，ハッシュ関数の性質から入力値（ナンスの値）が変化するだけでも，出力させるハッシュ値は大きく変動する．そのため，マイナー，一定の値よりも小さくするため，ナンスの値を推測することは基本的にできません．従って，PoW を行い報酬をもらうマイナーは，ナンスの総当り計算をするしか方法がない条件（難易度）に合うナンスを見つけるとマイニング成功となり，成功したマイナーはマイニング報酬を受け取ることできる．

ブロックに対してハッシュ値を計算する際に，先頭から特定の個数 0 が並ぶようなハッシュ値を，ナンスという値を変えながら探していくという作業である．ハッシュ値の計算は，逆算ができない性質を持つことにより，特定の性質を持つハッシュ値となるよう 1 つ 1 つ計算して見つけるしかない．

従って，非常に大規模な計算が必要となる．この作業により見つけられた先頭から特定の個数 0 が並ぶハッシュ値を持ったブロックだけをネットワーク全体で「正しいブロック」として認める．ハッシュ値の先頭から並ぶ 0 の数は，多ければ計算が難しく，少なければ容易となる．ビットコインでは，およそ 10 分に 1 回の頻度で条件を満たすハッシュ値を見つけられる程度の個数に調整されている．これは，コンピュータの性能向上やユーザ数の増加に応じて変更される．

PoW の大まかなプロセスは以下の通り

- ネットワーク上を伝搬する取引データを受信して記録しておく．
- トランザクションをまとめてブロックを生成する．
- ナンスの値を少しずつ変えながら大量のハッシュ計算しナンス値を探索を行う．
- 条件に合うナンス値を見つけた人がブロックを完成させてほかのノードに報告する．

- ほかのノードはブロックの中身とナンス値が正しいかどうか検証する.
- ナンス値が正しければ，ブロックを完成させた人に報酬が支払われる.

difficulty ターゲットを満たす SHA256 のアルゴリズムに対し，解を見つけなければなりません.

暗号学的ハッシュ関数

暗号の利用方法の 1 つに，メッセージの完全性を確認するための情報であるメッセージダイジェストと呼ばれる固定長のビット列を生成するものがある．任意の長さのメッセージから固定長のビット列を生成する暗号アルゴリズムは，暗号学的ハッシュ関数と呼ばれる.

暗号学的ハッシュ関数は，以下の性質を持つ.

- 原像計算困難性
- 第 2 原像計算困難性
- 衝突困難性

原像計算困難性とは，ハッシュ値 h が与えられたとき，$h = hash(m)$ となるような任意のメッセージ m を探すことが困難な性質である.

第 2 原像計算困難性とは，入力 $m1$ が与えられたとき，$hash(m1) = hash(m2)$ となるようなもう 1 つの入力 $m2(m1$ とは異なる入力) を見つけることが困難である性質である.

衝突困難性とは，$hash(m1) = hash(m2)$ となるような 2 つの異なるメッセージ $m1$ と $m2$ の組を探すことが困難である性質である．これらの性質によって，2 つのメッセージに対するダイジェストが同じである場合，メッセージが同一である可能性が高いことを意味する.

また，逆にダイジェストが異なる場合は，メッセージは異なるものであることを意味する.

文書の完全性を保証する情報としては，メッセージダイジェストが用いられる．メッセージダイジェストは入力文書の大きさのよらない固定長の情報（ハッシュ値）であり，以下の特徴を持つ.

- メッセージからメッセージダイジェストが簡単に計算できる.
- メッセージダイジェストからメッセージを見つけることは出来ない.
- メッセージから同じメッセージダイジェストが生成される別のメッセージを見つけ

ることはできない.

- メッセージを１ビットでも変更すれば，メッセージダイジェストは全く異なる値になる.

このような特徴を持つメッセージダイジェストを生成するアルゴリズムとして，$MD5$, $SHA-1$ などのハッシュアルゴリズムが従来から使われてきたが，$MD5$ はすでに破られており，$SHA-1$ についても脆弱性が指摘されている．現在は，$SHA-2(SHA-256)$ あるいはその後継として公募された $SHA-3$ への移行が急がれている.

ビットコインにおけるブロック構造

ビットコインブロックチェーンにおけるブロック構造は以下図のように構成されている.

Version ソフトウェア／プロトコルのバージョン番号 Previous Block Hash 親ブロック（１つ前のブロック）のハッシュ値 Merkle Root ブロック内の全トランザクションに対するマークルツリーのルートハッシュ Timestamp ブロックの生成時刻 Difficulty Target ブロック生成時の Proof of Work の difficulty Nonce Proof of Work で用いるカウンタ

- transaction（トランザクション）
- Previous Block Hash
- Merkle Root
- Timestamp
- Difficulty Target
- Nonce

ハッシュレート

本研究での，ハッシュレートとは，単位時間あたりに暗号学的ハッシュ関数の計算を行うことができる回数である．この時の改ざん耐性は，マイニングに勝利してブロックを生成するのは，ハッシュレートが最も能力のあるノードである為，ハッシュレートの最大値のノードが基本的にマイニングに勝利し報酬を受け取ることができる.

公開鍵暗号方式

公開暗号方式では，暗号化と復号でそれぞれ異なる鍵を使う．秘密鍵と公開鍵が利用され，秘密鍵は自分だけが持っており他者に教えてはいけない鍵．公開鍵は広く公開している鍵とそれぞれの役割が異なっている．ここで，秘密鍵と公開鍵の重要な特徴として，公開鍵から秘密鍵が逆算されないことが挙げられ，そうでなければ，人に教えてはいけない情報を，誰もが知っている公開鍵から特定されてしまい，暗号技術として成り立たなる．ビットコインブロックチェーンでは公開鍵暗号方式が使われており，アドレスの生成や取引データのやり取りなど，様々な場面で利用されている．

共通暗号方式

共通暗号方式は，暗号化と複合で同じ鍵を用いるため，鍵を安全に相手と共有する必要がある．また共有する相手の数だけ鍵が必要なため相手が増えれば増えるほど管理が煩雑になるデメリットがある．このデメリットを回避するために，公開暗号方式が開発された．

楕円曲線暗号

楕円曲線暗号は１９８０年代半ばに提案された暗号方式で楕円曲線という特殊な曲線を利用している．暗号アルゴリズムはコンピューターの処理性能の向上や攻撃手法の高度化によって日々脅威にさらされている．そのため，利便性を保ちつつ，高い安全性を維持する方法が常に検討されている．

3.4 スケーラビリティ問題へのアプローチ方法

スケーラビリティの問題へのアプローチとしていくつか研究として行われている．

3.4.1 オフチェーン（off-chain）

Lightning Network[6] のようなオフチェーンスケーリング技術が注目されることが多い．しかし，オフチェーン取引はブロックチェーン上に記録を残さないため，監査可能性を最大化するというビットコインの本来の考えから遠ざかっている．

3.4.2 オンチェーン (on-chain)

オンチェーンのスケーラビリティ問題を解決するためには，ブロックサイズを大きくするか，ブロック生成の時間間隔を短くする 2 つのアプローチで行われている．

3.5 ブロックチェーンコアノードのドメイン分割

ブロックチェーンコアノードの分割は，文献 [2][3] で提案されている．この提案手法に基づいて本研究は進める．地理的に近いコアノード間で自律的にドメインを形成し，ドメインごとに固有のブロックチェーンを管理する方法が提案されている [7, 8, 9]．このしかし，各ドメインのコアノード数が減少すると，ブロックチェーンの分散性や改ざん耐性が低下するという問題がある．

電子署名

通常の電子署名は秘密鍵が漏洩することによる改ざんが可能であり，改ざんされたことの検知も不可能である．

一方，ヒステリシス署名では，電子署名が入れ子構造になって署名がされている為，署名の改ざんを行う為には，改ざんした内容以降の全ての入れ子構造になった署名を改ざんする必要があり，改ざんが著しく困難になる．また，部分的に改ざんされた場合は，入れ子構造を検証することで，改ざんの検知も可能になる．これは PoW に代わる改ざん耐性技術としてブロックチェーンが注目を集める前から知られており，過去のデータ等を改ざん耐性を持たせて保存することができる．

3.6 ヒステリシス署名

ヒステリシス署名は，過去の署名履歴を取り込みながら署名生成を行う方式である．

ヒステリシス署名 [4] とは，通常の電子署名に連鎖する入れ子構造を持たせることで，その改ざん耐性を向上させる技術である．図 5 にヒステリシス署名の構造を示す．1 つ前のヒステリシス署名 S_{n-1} を暗号学的ハッシュ関数にかけたハッシュ値 $H(S_{n-1})$ と一緒に保存したい m 個のデータ $(B_{D_1}, \cdots, B_{D_m})$ のハッシュ値 $(H(B_{D_1}), \cdots, H(B_{D_m}))$ を追加する．この全体の署名 S_n を更に追加して生成されたものがヒステリシス署名となる．

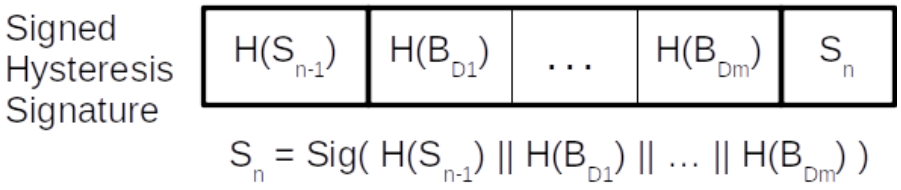

図 5　ヒステリシス署名の例

これにより，有効期限のが切れた電子証明書に対応する電子署名であっても，有効期限内の電子署名書により真正性が保証された電子署名空から連鎖構造を辿ることで，結果として真正性を証明可能となる．

なお通常の署名方式では，署名方式では，署名対象となるメッセージと電子署名とを組みにして相手の（検証者）に送るが，ヒステリシス署名では，それに加えてひとつ前の署名結果のハッシュ値も合わせて送ることが可能になる．

時間情報を文書に織り込むのではなく，署名者が過去に関わった署名付き電子文書の圧縮データを次々に織り込んで行くことである．

3.7　履歴交差法

ヒステリシス署名の証明力はこうしたヒステリシス署名つき文書が作成したエンティティの電子文書が署名を作成したエンティティの手を離れ，他のエンティティの電子文書におけるヒステリシスとして署名に織り込まれた時に，さらに飛躍的に向上すると考えられる．

3.8　Raft Consensus Algorithm

[11] 複数のノード間で合意を得る必要がある．分散システムで合意するを得る手段に Raft がある Raft は Paxos ど同等の性能を持ちながら Paxos よりも格段の理解のしやすさを提供する分散合意プロトコルである．Raft のアーキテクチャ概要を図 1 に示す．Raft を実行するノードには, LEADER，CANDIDATE，FOLLOWER という 3 種類の状態がる．LEADER はクラスタ内に 1 台しか存在しない．クライアントは，LEADER ノードとインタラクションする．クライアントは LEADER にアクセスしてきた場合は LEADER にリダイレクトされる．FOLLOWER のノードは LEADER ノードに従う．LEADER ノードはクライアントノードから更新要求を受信すると，ログレコードを作成する．そして LEADER ノードはログレコードをストレージに保存した後，ログレコー

ドをログレコードを FOLLOWER ノードへ転送する．FOLLOWER ノードは受信したログレコードをストレージに保存すると，ACK を LEADER へ送信する LEADER を含め，Raft クラスタの過半数が命令を受け取ったことを LEADER が確認した後，クライアントに COMMIT を通知する．最終的には，すべてのノードが同じ命令を同じ順番で実行するため，すべてのノードでデータが複製される．即ち Raft は複製化状態機械に基づいて設計されている．一般に利活用されている複製化状態機械に例として，Chubby, ZooKeeper などが挙げられる．LEADER ノードが故障したり，LEADER ノードの故障したり，LEADER ノードのネットワークコネクションが切断されると，Raft クラスタ内に LEADER が存在しなくなる．このような場合，FOLLOWER ノードは状態を CANDIDATE へと変更する．クラスタ内のノードは各 CANDIDATE への投票を行い，新たな LEADER を選出する．この選出にはクラスタを構成するノードの数の過半数の選任が必要である．過半数を得たノードが存在しなければ，LAEDER 選出作業を再実行する．

4　ブロックチェーン履歴交差法

提案するブロックチェーン履歴交差法について説明する．

4.1　提案手法

本研究において提案する P2P ネットワークの構成を図 **??**に示す．

本研究における P2P ネットワーク環境は，Layer0 と 1 のネットワーク層から構成される．Layer1 の層は通常のブロックチェーンを共有する P2P ネットワークである．従来のブロックチェーンを共有している P2P ネットワークに所属するノードの集合のことをドメインとして位置づけている．本研究では，複数のドメインが互いにブロックチェーンに「指紋」としての役割を持つヒステリシス署名として共有する履歴交差法行うことで，全ドメインの改ざん耐性での改ざん耐性向上を目指す．

Layer1 では，各ドメイン $(D_1, D_2, \cdots, D_m)$ が独自のブロックチェーンを管理する．Layer0 は，Layer1 のドメインを代表する中心コアノード同士が Layer1 とは異なる，Layer0 の P2P ネットワークにより相互接続している．

4.1.1　本研究におけるブロック構造

提案するブロックの構造を図 7 に示す．

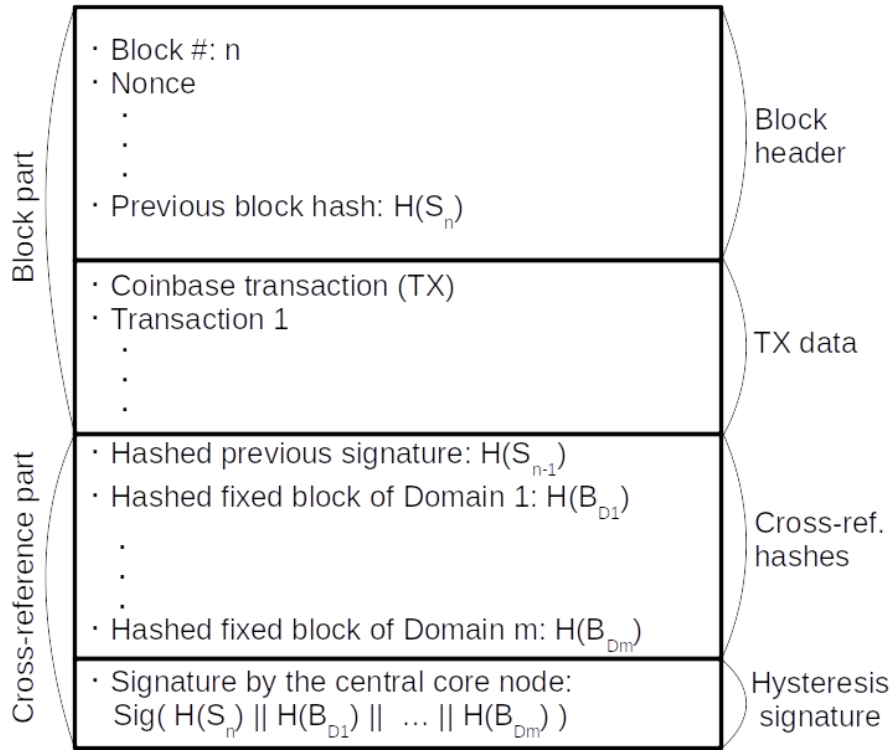

図 7　提案するブロック構造

従来のブロック構造に履歴交差部 (Cross-reference part) を追加する．履歴交差部に書き込むヒステリシス署名は，Layer0 を介して共有する．

4.1.2　履歴交差法

履歴交差法の動作アルゴリズム 1 を図 8 に示す．

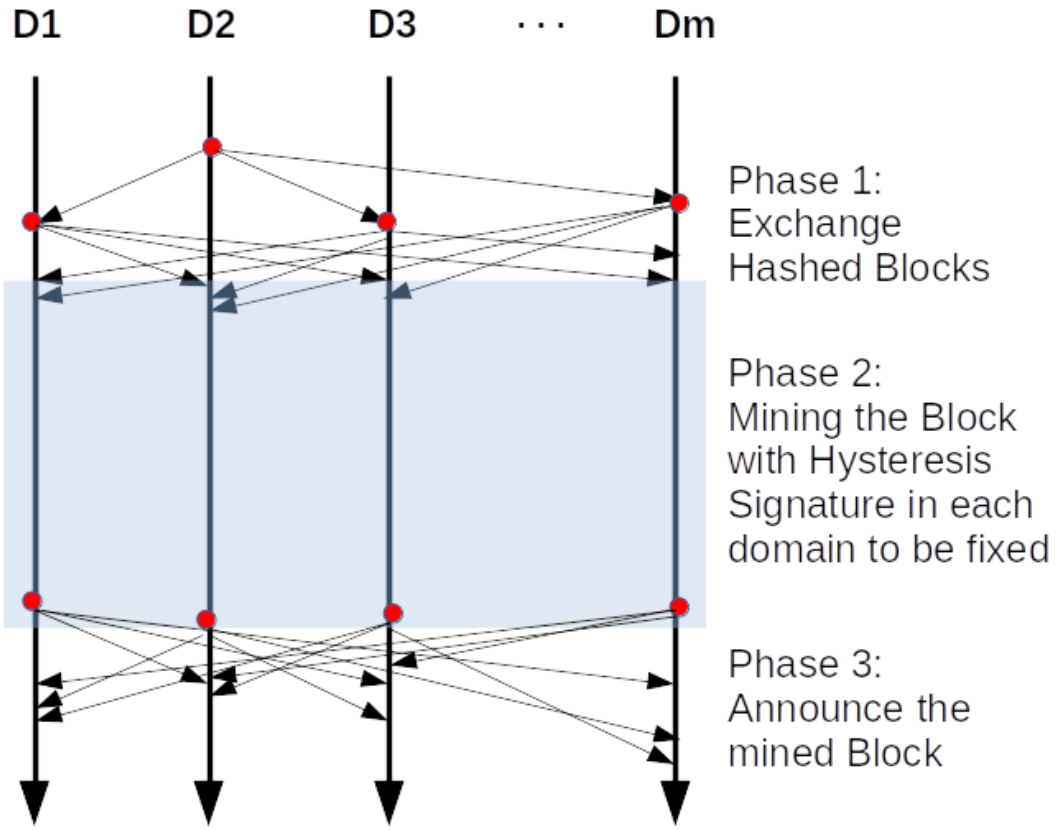

図 8　提案する履歴交差法

全体の動作は，次の 3 つの Phase から構成される．以下にその流れを簡単に説明する．

1. Phase1 では，ドメイン間で l-確定ブロック（最新のブロックから l 個前のブロック）を含むメッセージをドメインを代表する中心コアノード同士で互いに送り合う（ただし l は正の整数）．
2. Phase2 では，各ドメインが届いたメッセージを用いて図 5 のヒステリシス署名を行い，各ドメインが独立に署名を履歴交差部に書き込む為のマイニングを行う．（従来のマイニングと同様）
3. Phase3 では，マイニングし終えたブロックが，l-確定ブロックになったことを他のドメインにブロードキャストして報告する．

以上を提案する履歴交差法の一連の流れとし，履歴交差法を各ドメイン間で自律的に行う．

履歴交差法を実現する分散アルゴリズム [8] を設計した．その詳細を図 9，10 に示す．本研究における履歴交差を実現する 2 つのアルゴリズムが正しく動作する前提条件として，次の 1〜3 が挙げられる．

1. 中心コアノード同士が作る P2PNW は同期システムである．また，その構造は完全グラフとする．
2. 中心コアノードは正しくアルゴリズムを実行し，互いに必ずブロックを転送する．
3. アルゴリズム 1 では，全ての中心コアノードに故障停止がない場合に動作する．
4. アルゴリズム 2 では，t-故障停止がある場合を想定する．つまり，最悪 t 個の中心コアノードが故障停止に陥った場合でも動作する．

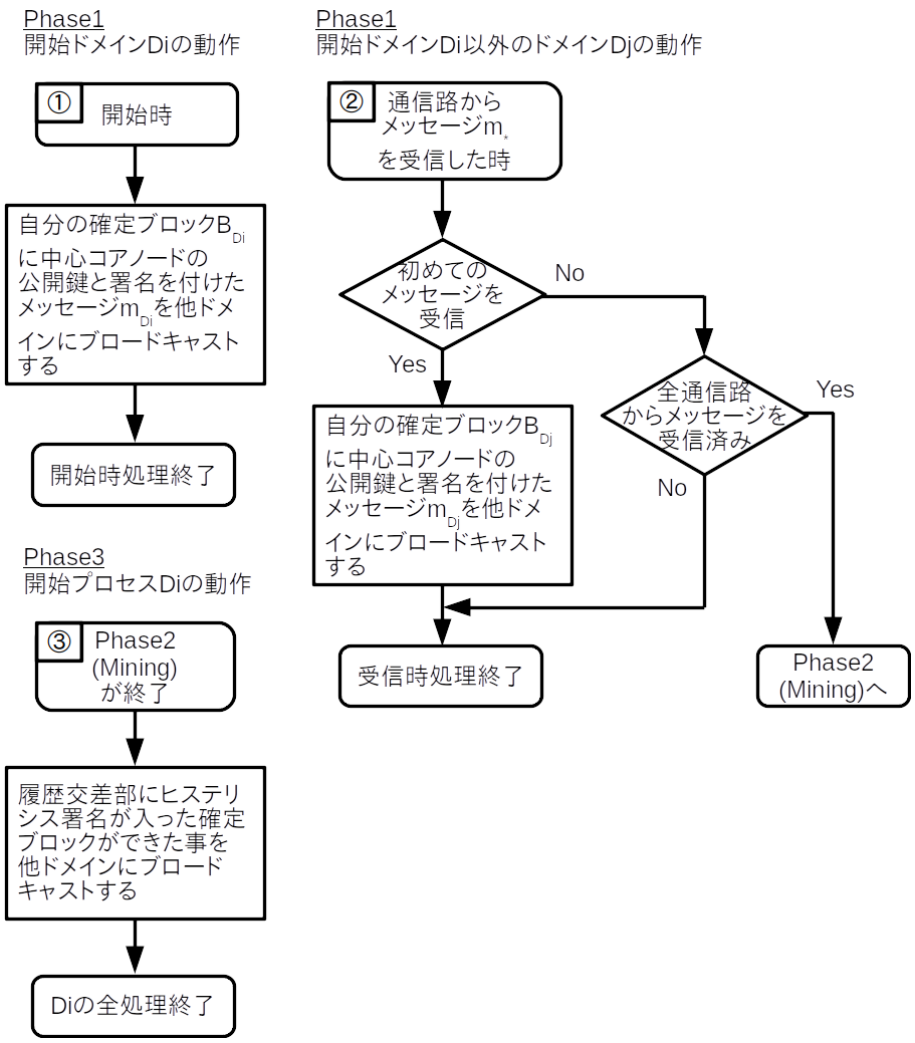

図 9　アルゴリズム 1（Phase 2 は Layer1 における通常のマイニング作業を行うだけである為，省略した.）

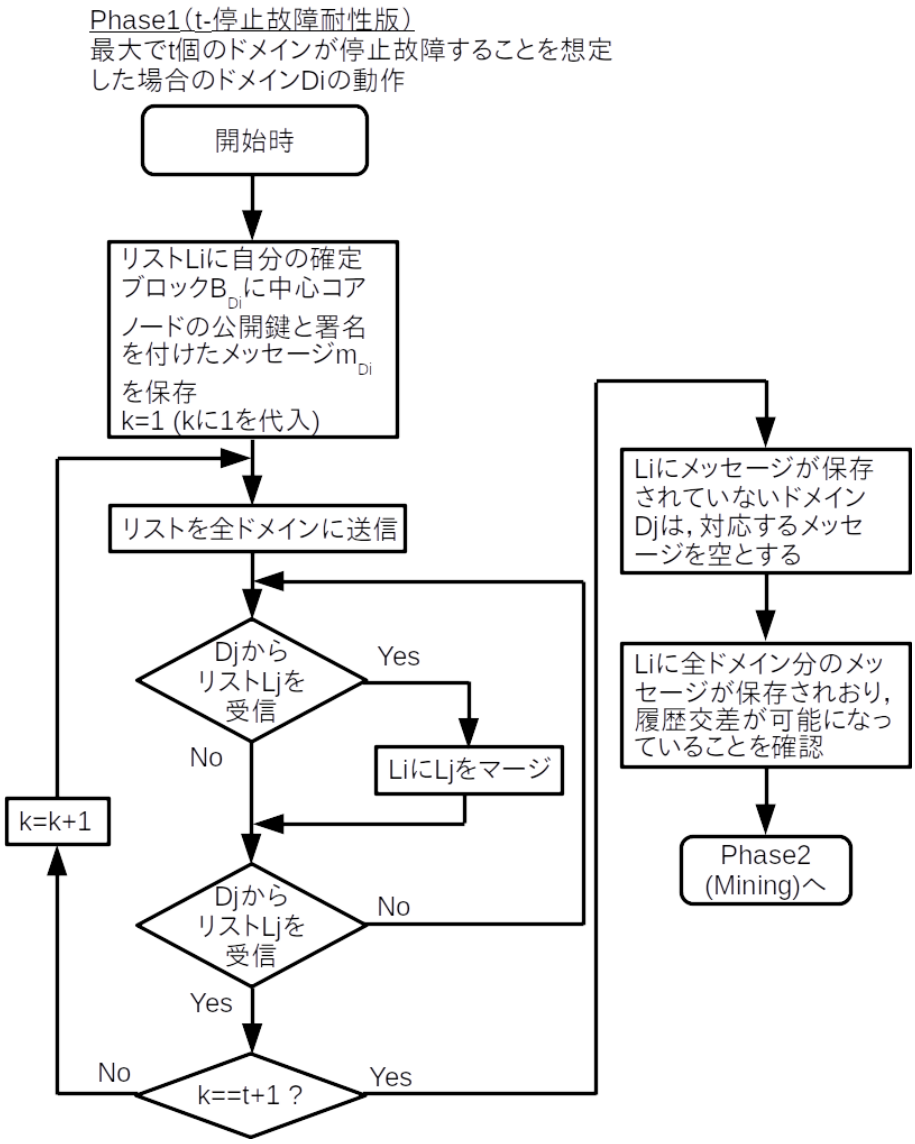

図 10　アルゴリズム 2(Phase 1 のみを記述．その他の Phase はアルゴリズム 1 と共通．)

4.1.3　提案方式の分散アルゴリズムの効率

通信計算量のとは，ある問題を解くアルゴリズムが開始してから終了するまでに送信されるメッセージの総数の最大値である．最大値を評価する以外に，平均値での評価やメッセージの総数ではなくメッセージの総ビット数で評価することもある．最大値で評価するのは最悪のケースを想定することを意味する．　非同期な分散システムでは，計算が開始してから終了するまでの時間を評価することができない．そこで，通信の遅れがプロセス

内での計算よりも短いという仮定のもとで，次のような仮想的な同期実行を考える．『第１ラウンドにおいて，各プロセスは同時に実行を開始し，内部計算およびほかのプロセスへの送信を実行し，ほかのプロセスからのメッセージ受信待ちになるまで動作を行う．第 $n+1$ ラウンドでは，第 n ラウンドで送信されたメッセージがすべて受信側プロセスに到着する．各プロセスはそれらのメッセージを受信し，受信したあとに実行する内部計算およびほかのプロセスからのメッセージ受信待ちになるまで動作を行う.』このように動作を行った場合のラウンド数を時間計算量という．時間計算量という．同様に時間計算量においても，最悪値や平均値で評価する．

分散アルゴリズムの効率は，通信計算量と時間計算量で評価することが一般的である．提案した手法の分散アルゴリズム 1，2 の効率は，アルゴリズム 1 が開始してから終了するまでに送信される通信計算量は $O(m^2 \cdot b)$，アルゴリズム 2 の通信計算量は $O((t+1)(m^2 \cdot b))$ となる．また動作を行った場合の時間計算量は，アルゴリズム 1 は，$T_1 + T_2 + T_3$，アルゴリズム 2 は $(t+1)T_1 + T_2 + T_3$ となる．ここで $T_i (i = 1, 2, 3)$ は，アルゴリズム 1 において Phase i にかかる時間である．

4.2 履歴交差法の理論的性能

履歴交差法を行うことで，ブロックチェーンの改ざん耐性が向上することが期待される．全世界のコアノードで管理するビットコインブロックチェーン方式と近接するコアノードでドメインを形成位し，履歴交差法を用いた方式で改ざん耐性向上比率を定義し，理論的にその比率を評価する．全世界ノード数 N で，ノード i のハッシュレートを h_i とする．この時の改ざん耐性は，マイニングに勝利してブロックを生成することができるのは，ハッシュレートが最も大きなノードである為，ハッシュレートの最大値与えられる．

$$\max(h_1, h_2, \cdots, h_N) \tag{1}$$

ドメイン数が m 個で，各ドメインに所属するノードの数が D_m 個の場合を考える．各ドメインの改ざん耐性は，ドメインに属するノードのハッシュパワーの最大値で与えられる．

$$A_1 = \max(h_{11}, \cdots, h_{1D1}), \tag{2}$$
$$A_2 = \max(h_{21}, \cdots, h_{2D2}), \tag{3}$$
$$\vdots$$
$$A_m = \max(h_{m1}, \cdots, h_{mDm}) \tag{4}$$

このドメインの中でハッシュレートが最大のノードがドメインが履歴交差法に参加するインセンティブとなるように，全ドメインの中でハッシュパワーが最大のものと比較する．

$$A = \max(A_1, A_2, \cdots, A_m) \tag{5}$$

上記の m 個のドメインで履歴交差法を実行すると，全てのドメインのブロックチェーンを改ざんしない限り，履歴交差部にヒステリシス署名の証拠が残ってしまう為，改ざんが検出されてしまう．従って，改ざん耐性は全ドメインの最大ハッシュレートの和で表される．

$$B = \sum_{i=1}^{m} A_i \tag{6}$$

以上より，ドメイン i が履歴交差法に参加することで，期待される改ざん耐性向上比率 R_i は次式で表される．

$$R_i = \frac{B}{A_i} > 1 \quad (i = 1, \cdots, m) \tag{7}$$

また，ハッシュレートが最大のドメインが履歴交差法に参加することで期待される改ざん耐性向上比率 R は次式で表される．

$$(R_i \geq) R = \frac{B}{A} > 1 \quad (i = 1, \cdots, m) \tag{8}$$

このように，理論的には全てのドメインで改ざん耐性向上率を見積もることができる．

ここで，改ざん耐性向上率 R を見積もる為に，ハッシュレートをランダムに割り振って乱数シミュレーションを行うことにより，数値的に評価する．全ノード数を 1 万とし，ドメイン毎に分割されたノード数を一律 10, 100, 1000 ノードとした場合を評価した．全ノード数を 1000 万ノードとし，ドメイン毎に分割されたノード数を一律 10, 100, 1000 ノードとした場合のハッシュレート h_{ij}（i:ドメイン番号，j:ドメイン内のノード番号）を次式であわらされるパレート分布とした．

$$P(h_{ij}) = \frac{\alpha}{h_{ij}^{1+\alpha}} (h_{ij} > 1). \tag{9}$$

ここで，α はパレート分布のスケールパラメータである．この値が小さいほどハッシュレートに格差が大きくなる．一般にハッシュレートは CPU の数に比例する．CPU をいくつ購入できるかは，ノードを所持する人物ないしはグループの資本と関係している為，

富の分布と相関があると予想できる．この理由により，ハッシュレートの分布をパレート分布とすることは妥当であると考える．シミュレーションの実行結果を図 11, 12 に示す．

一般にドメイン数が増加するほど，改ざん耐性向上比率 R の分布のピークが大きい方にシフトすることが分かる．また，ドメイン数が増えると分布の分散も大きくなることが確認できる．また図 11 と図 12 の比較により，スケールパラメータ α が小さくなるほど，改ざん耐性向上比率 R の分布のピークが小さい方にシフトすることも分かる．

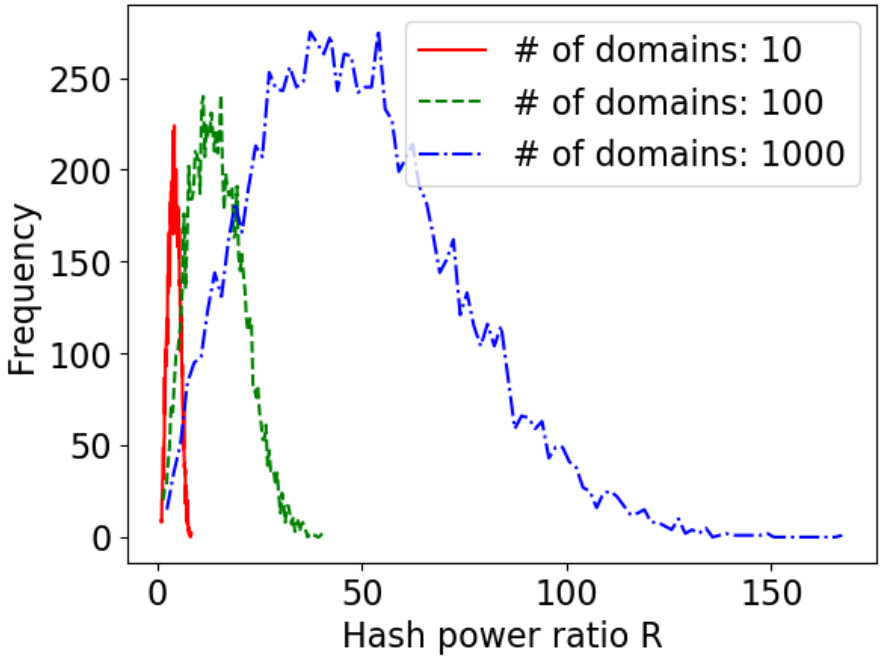

図 11　改ざん耐性向上比率 R の確率分布（ $\alpha = 2$ の場合 ）

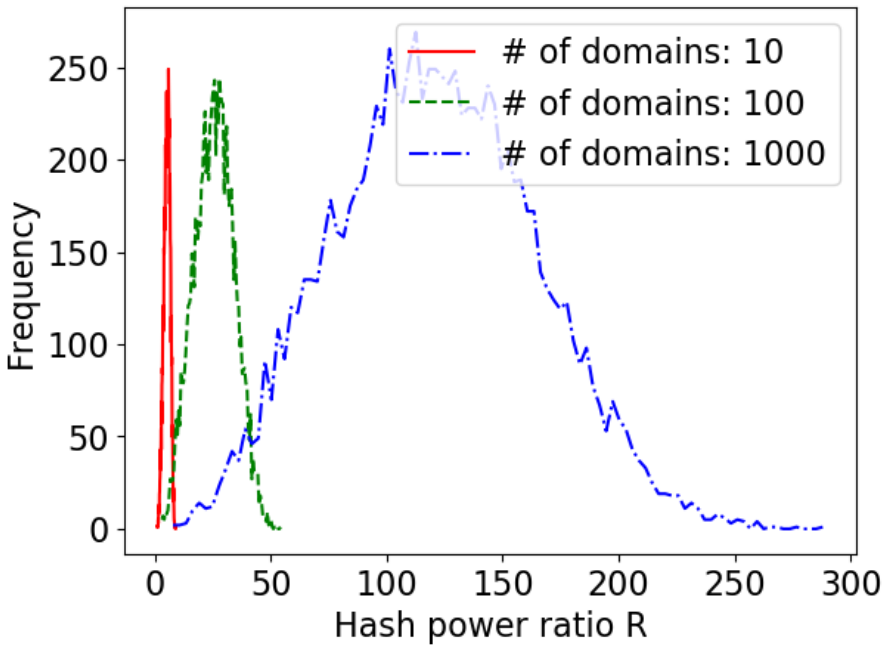

図 12　改ざん耐性向上比率 R の確率分布（ $\alpha = 3$ の場合 ）

5 実装と実験

5.1 実装

提案する手法を実現するコアノード実装に当たり Layer0 における通信プロトコルの設計を行った.

5.1.1 通信プロトコルの実装

Layer0 における中心コアノード同士が履歴交差法を行うための実装した通信プロトコルを以下に示す.

1. **MSG_REQUEST_CROSS_REFERENCE**
   （履歴交差の依頼 ）
2. **MSG_ACCEPT_CROSS_REFFERENCE**
   （履歴交差依頼の承認 ）
3. **MSG_START_CROSS_REFFERENCE**
   （履歴交差の開始号令 ）
4. **MSG_CROSS_REFFERENCE**
   （履歴交差を行いたい block 内容の共有 ）
5. **MSG_COMPLETE_CROSS_REFERENCE**
   （履歴交差の完了通知 ）

4.1 節の提案手法の実験を行う為に，実際に履歴交差法を行うことができるノードを実装した．実装するにあたり，文献 [9] を参考にした.

以下に実装した通信プロトコルを示す.

1. **MSG_REQUEST_CROSS_REFERENCE (MSG_1)**
   （履歴交差の依頼 ）
2. **MSG_ACCEPT_CROSS_REFFERENCE (MSG_2)**
   （履歴交差依頼の承認 ）
3. **MSG_START_CROSS_REFFERENCE (MSG_3)**
   （履歴交差の開始号令 ）

4. **MSG_CROSS_REFFERENCE (MSG_4)**
（履歴交差を行いたい block 内容の共有 ）
5. **MSG_COMPLETE_CROSS_REFERENCE (MSG_5)**
（履歴交差の完了通知 ）

ローカル環境に構築した環境において実験を行った．今回は簡単の為に，特定（ドメイン D_2）の中心コアノードが 2 分間隔で履歴交差の依頼を他ドメインの中心コアノードに行い，履歴交差を開始させることとした．

5.2 実装した履歴交差法の動作

Phase1 では，ドメイン D_2 の中心コアノードが履歴交差の依頼 (**MSG_1**) を他ドメインの中心コアノードに送信し，ドメイン D_2 以外の中心コアノードが受信する．依頼を受信した各ドメインは履歴交差の依頼を了承する返信 (**MSG_2**) をドメイン D_2 に送信する．ドメイン D_2 の中心コアノードは，他の全てのドメインから了承の返信を受け取ったら，履歴交差を開始する号令 (**MSG_3**) を他ドメインの中心コアノードに送信し，履歴交差を行いたい l-確定ブロックのデータを各ドメインの中心コアノードにブロードキャストする (**MSG_4**)．同様に号令を受け取った D_2 以外の中心コアノードは，各ドメインの中心コアノードにブロードキャストを行う．個々のドメインの中心コアノードは全てのドメインからの確定ブロックのデータを受取り，データを暗号学的ハッシュ関数にかけて hash 値にする．以上で Phase1 は完了となる．

Phase2 では，Phase1 の履歴交差部を書き込む為のマイングを行う．本研究におけるマイニングは，ビットコンブロックチェーンにおける PoW と同様のものである．今回の実験では，PoW の難易度を 5 とした．今回実験を行ったマシンでは，この難易度でブロックを生成するのに 10 秒程度の時間がかかっている．履歴交差を行ったブロックが l-確定ブロックとなった時点で Phase2 は完了となる．

最後に Phase3 では，マイニングを完了したブロックが，履歴交差を行ったブロックから l-確定ブロックとなった段階で，他ドメインの中心コアノードに履歴交差に成功したことをブロードキャストする (**MSG_5**)．今回の実験では，$l = 3$ とした．ブロードキャストを送信した時点で Phase 3 は完了となる．

5.3 通信遅延の考慮

ドメインの最小規模の 1 国 2〜4 ドメイン程度で分割することを想定している．従って，ドメイン間の通信は，他国間の通信の想定をする必要がある．今回の実装には，多国間で通信遅延を考慮し通信遅延処理を追加し実装した．

5.3.1 疑似通信遅延機能の実装

各ドメインが Balcelona，Tokyo，Paris，Toronto，Washington にあるとし，履歴交差の依頼を行うドメイン D_2 は Tokyo のものとした．2020 年 2 月 9 日の ping の RTT にかかる時間 [10] を参考にすることで，送信先のポート番号で送信先ドメインを区別して，sleep 関数を利用することで遅延を発生させた．

参考にした文献 [10] は，unix コマンドラインツールの ping を用いており，ウインドウサイズを今回は 32[Byte] とすると，ping の RTT を P[ms] から，送信データサイズを B[Byte] に応じた通信遅延 L は，以下の式で表される．

$$L = \frac{BP}{32}[sec] \tag{10}$$

送信するデータサイズと送り先のポート番号に応じて通信遅延 L を発生させる処理とした．通信する際におおよそ 0.1〜2 秒程度の通信遅延 L が発生している．実際に参照した遅延時間を図**??**，算出した Tokyo から 150〜500[Byte] の疑似データサイズを他のドメインへの送信した際の通信遅延 L の上限値を表 1 に示す．

表 1　Tokyo からの各ドメインへの疑似通信遅延 (小数第 2 以下は省略)

[sec]	Barcelona	Paris	Toronto	Washington
Tokyo	1.0〜3.8	1.0〜3.8	0.6〜2.4	0.8〜2.6

5.4 t-故障耐性の実装

本研究における故障は，フリーズ等の機器故障を故障状態として定義している．履歴交差法を行う際に Layer0 における各ドメインの中心コアノード以下の状態で故障することがある．

1. 履歴交差の依頼を出し履歴交差法の号令を取る中心コアノードの故障（状態 A）

2. 履歴交差の依頼を受け，履歴交差に参加するノードの故障（状態 B）

履歴交差の依頼を出す中心コアノード（状態 A）が故障してしまうと履歴交差の依頼を出すノードが不在になり履歴交差法を実行ステップに移行できなくなる．状態 A の統率役不在（故障）時を回避するために 3.8 節の Raft Consensus Algorithm3.8 を用いる．

履歴交差の依頼を受け，履歴交差に参加するノードの故障（状態 B）が，故障してしまった場合はリーダー権を保持しているノードがタイムアウトで履歴交差から除外を行う．

5.4.1 実験結果

??節で実装したプロトコルを用いて，履歴交差を 1000 回行い，各 Phase が終了するまでにかかる遅延時間分布と履歴交差法の成功率について調査した．前節で解説した通信プロトコルを用いて実験を行った結果として，各 Phase での遅延時間の分布を図 14, 15, 16 に示す．

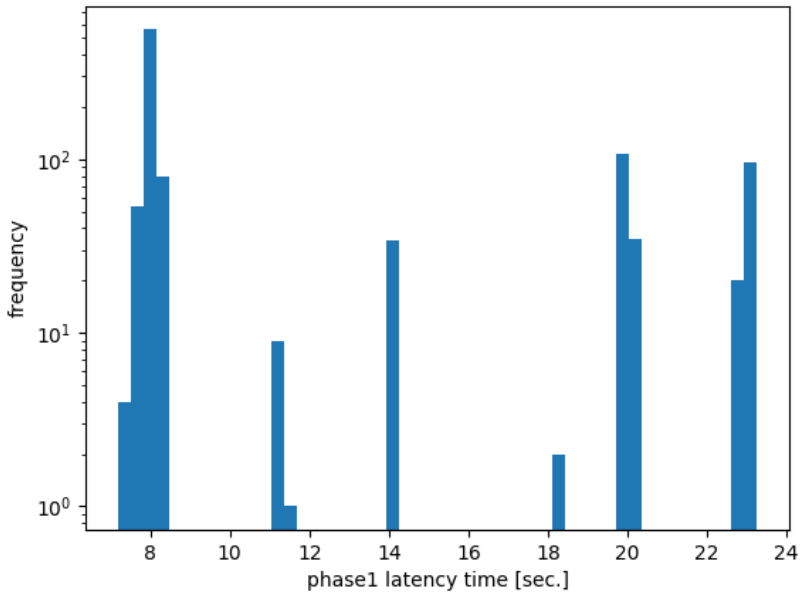

図 14　Phase1 の計測時間

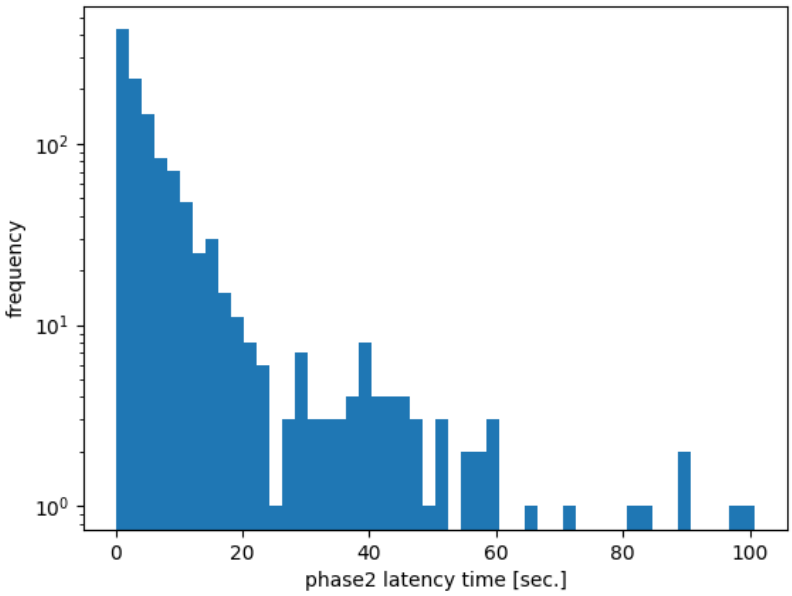

図 15　Phase2 の計測時間

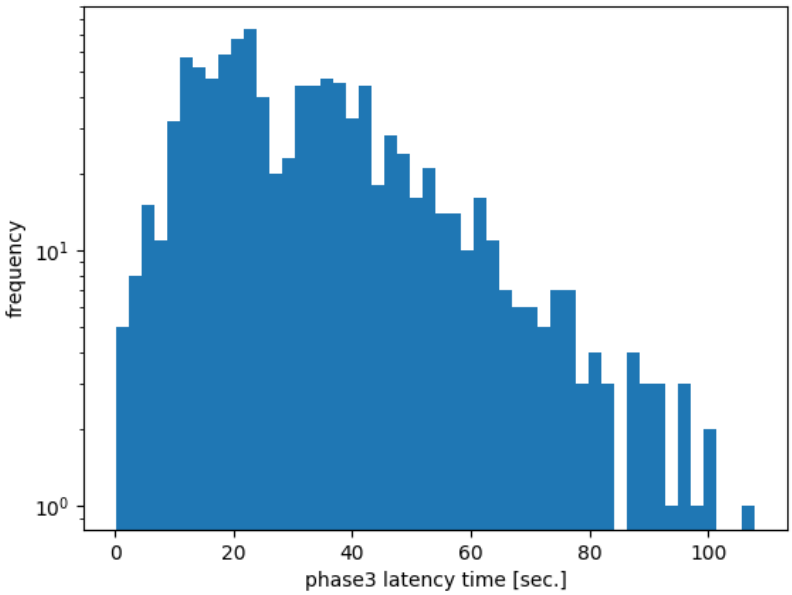

図 16　Phase3 の計測時間

Phase 1 は殆どが 8 秒以下で終わっているが，場合によっては最悪 20 秒程度かかることもあることが分かる．Phase 2 はマイニングを行うことから，指数分布に従っていることが分かる．Phase 3 は全てのノードのマイニングが完了するのを待つことから，タイミングが悪い場合は 100 秒程度待つ必要があることが分かる．全ての Phase を合計すると 120 秒程度になることから，履歴交差にかかる時間を見積もることができた．

また，1000 回の履歴交差のうち，失敗は 0 回であった．従って，成功率は 100% となった．

プロトコルを用いて，履歴交差を 1000 回行い，各 Phase が終了するまでにかかる遅延時間分布と履歴交差法の成功率について調査した．ここで，Layer 0 である中心コアノードをドメイン D_1, D_2, D_3, D_4, D_5 からそれぞれ 1 つの 5 ノードからなる状況を想定し，それらの間に P2P ネットワークをローカルに構築した環境において実験を行った．今回は簡単の為に，特定（ドメイン D_2）の中心コアノードが 2 分間隔で履歴交差の依頼を他ドメインの中心コアノードに行い，履歴交差を開始させることとした．

Phase1 では, ドメイン D_2 の中心コアノードが履歴交差の依頼 (**MSG_REQUEST_CROSS_REFE**
を他ドメインの中心コアノードに送信し，ドメイン D_2 以外の中心コアノードが受信する．依頼を受信した各ドメインは履歴交差の依頼を了承する返信 (**MSG_ACCEPT_CROSS_REFFERENCE**) をドメイン D_2 に送信する．ドメイン D_2 の中心コアノードは，他の全てのドメインから了承の返信を受け取ったら，履歴交差を開始する号令 (**MSG_START_CROSS_REFFERENCE**) を他ドメインの中心コアノードに送信し，履歴交差を行いたい l-確定ブロックのデータを各ドメインの中心コアノードにブロードキャストする (**MSG_CROSS_REFFERENCE**)．同様に号令を受け取った D_2 以外の中心コアノードは，各ドメインの中心コアノードにブロードキャストを行う．個々のドメインの中心コアノードは全てのドメインからの確定ブロックのデータを受取り，データを暗号学的ハッシュ関数にかけて hash 値にする．以上で Phase1 は完了となる．

Phase2 では，Phase1 の履歴交差部を書き込む為のマイングを行う．今回の実験では，PoW の難易度を 5 とした．今回実験を行ったマシンでは，この難易度でブロックを生成するのに 10 秒程度の時間がかかっている．履歴交差を行ったブロックが l-確定ブロックとなった時点で Phase2 は完了となる．

最後に Phase3 では，マイニングを完了したブロックが，履歴交差を行ったブロックから l-確定ブロックとなった段階で，他ドメインの中心コアノードに履歴交差に成功したことをブロードキャストする (**MSG_COMPLETE_CROSS_REFERENCE**)．今回の実験では，$l = 3$ とした．ブロードキャストを送信した時点で Phase 3 は完了となる．

6 結論

本研究では，独自のブロックチェーンを管理する複数のドメイン間で定期的にブロックチェーンの状態を共有することで，互いのブロックチェーンにヒステリシス署名を書き込み合う履歴交差法を提案した．

提案手法について理論的および実験的観点から，改ざん耐性の向上比率について理論的に評価した．一般にドメイン数が増加するほど，改ざん耐性向上比率 R の分布のピークが大きい方にシフトすることが分かった．また，ドメイン数が増えると分布の分散も大きくなることが確認できる．スケールパラメータ $\alpha = 2, 3$ の比較により，α が小さくなるほど，改ざん耐性向上比率 R の分布のピークが小さい方にシフトすることも分かった．

次に提案した履歴交差法を行うために以下の実装を行った．

1. 履歴交差法を行うための通信プロトコルの設計・中心コアノードの実装

2. t-故障耐性法の検討・実装

3. Raft-Consensus-Algorithm による Layer0 のリーダー選定法の提案・実装

実際に実装を行ったノードで，履歴交差法にかかる通信遅延時間を計測し，その成功率も実験的に評価した．その結果，120 秒程度待つことで履歴交差法の全 Phase が完了することが見積もれた．また今回の実験特定の条件下ではあるが，履歴交差法の成功率は 100% となった．

また Raft のを用いたリーダーの選定の実装及び，t-故障耐性の実装も行った．

今後の課題としては，Raft のリーダーとして主張し続けるノード動作の考慮やドメイン分割の自律形成法の提案，海外のレンタルサーバー間で実際の通信遅延発生状況下実験することや，想定される様々な状況において安定して履歴交差法が行えるように実験を継続することがある．また，理論的に得られた改ざん耐性向上比率を実験的に評価することがある．

謝辞

本研究は，令和 2 年度修士研究として，千葉工業大学大学院工学研究科電気電子情報工学専攻准教授 藤原明広先生のもとで行ったものである．本研究を遂行するあたり，適切なご指導，御助言をして頂いた藤原明広先生に深く感謝の意を表する．同専攻教授 菅原真司先生，並びに同専攻教授 枚田明彦先生には本論文を査読していただき，有益なご意見，コメントを頂いた．ここに感謝の意を表する．また藤原研究室の各位には，研究遂行にあたり日頃より有益なご討論ご助言を戴いた．この機会に皆様に深く感謝の意を表す．

参考文献

[1] S.Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System” (2008). https://Bitcoin.org/Bitcoin.pdf （2020 年 1 月 7 日閲覧確認）

[2] A. Fujihara, “Proposing a system for collaborative traffic information gathering and sharing incentiviz by blockchain technology,” Advances in Intelligent Networking and Collaborative Systems, pp.170-182, Springer (2019).

[3] A. Fujihara, “PoWaP: Proof of Work at Proximity for a crowdsensing system for collaborative traffic information gathering,” Internet of Things, 100046, Elsevier (2019).

[4] 洲崎誠一，松本勉，電子署名アリバイ実現機構 -ヒステリシス署名と履歴交差- 情報処理学会論文誌，Vol.43, No.8, pp.2381-2393, (2002).

[5] K. Saito, T. Kubo, ”BBc-1: Beyond Blockchain One” (2018). https://beyond-blockchain.org/public/bbc1-design-paper.pdf （2020 年 1 月 7 日閲覧確認）

[6] bloXroute, https://bloxroute.com （2020 年 1 月 7 日閲覧確認）

[7] Atomic swap, https://en.Bitcoin.it/wiki/Atomic_swap#Algorithm（2020 年 2 月 13 日閲覧確認）

[8] 真鍋義文, 「分散処理システム」森北出版 (2013).

[9] 濱津誠, 「ゼロから創る暗号通貨」PEAKS 出版 (2018).

[10] WonderNetwork, https://wondernetwork.com/pings（2020 年 2 月 10 日閲覧確認）

[11]

付録

コード

コードを載せる.

付録

補足

補足説明があればのせる.