KARAKASA: 分散ハッシュテーブルを用いたBlockchainストレージのロードバランシング

2019.2.15

The 3rd Workshop Basing Blockchain

慶應義塾大学大学院 政策・メディア研究科

阿部涼介

chike@sfc.wide.ad.jp

発表概要

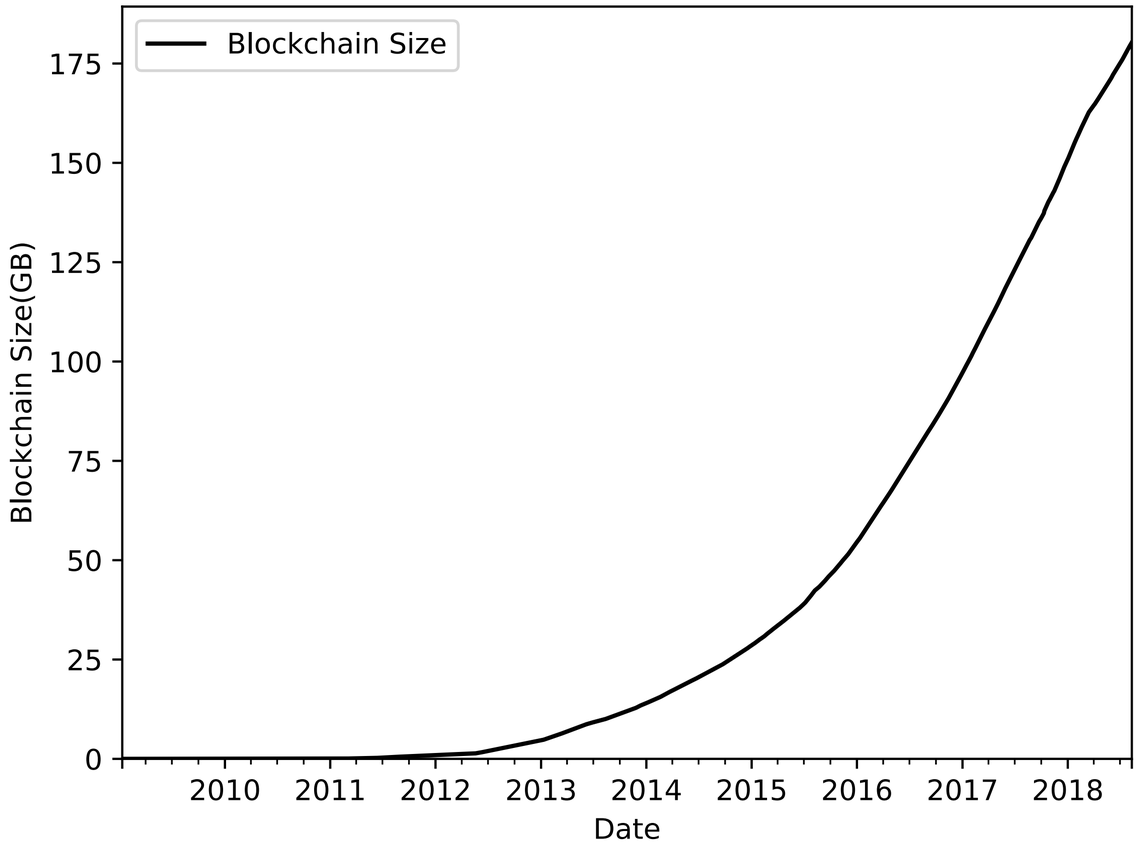
目的	リソースの限られたデバイスをBitcoinノードとして動作可能にする	
問題	十分なストレージ容量の見積もりは不可能	独立した検証の欠落
提案手法	KARAKASA: DHT(分散ハッシュテーブル)をベースにした分散ストレージを用いたBlockストレージの負荷分散スキーム	
解決手法	· 1ノードあたりの必要なストレージ容量の削減 · スケール可能なストレージ	· Blockchain全体へのアクセスが可能なことを維持

- Ryosuke Abe, “Blockchain Storage Load Balancing Among DHT Clustered Nodes” ArXiv, 2019, https://arxiv.org/abs/1902.02174
- Abe, Ryosuke, Shigeya Suzuki, and Jun Murai. "Mitigating bitcoin node storage size by DHT." Proceedings of the Asian Internet Engineering Conference. ACM, 2018.

Bitcoin概要

- **Bitcoin**
  - P2Pの暗号通貨
  - 支払いは「Transaction(TX)」で表現
  - 各ノードは新規TXの正当性を検証
  - 「Full Node」は公開台帳「Blockchain」に検証済みTXを全て保存

- **Blockchainデータ構造**
  - 追記専用
  - Blockchainサイズは増加し続ける

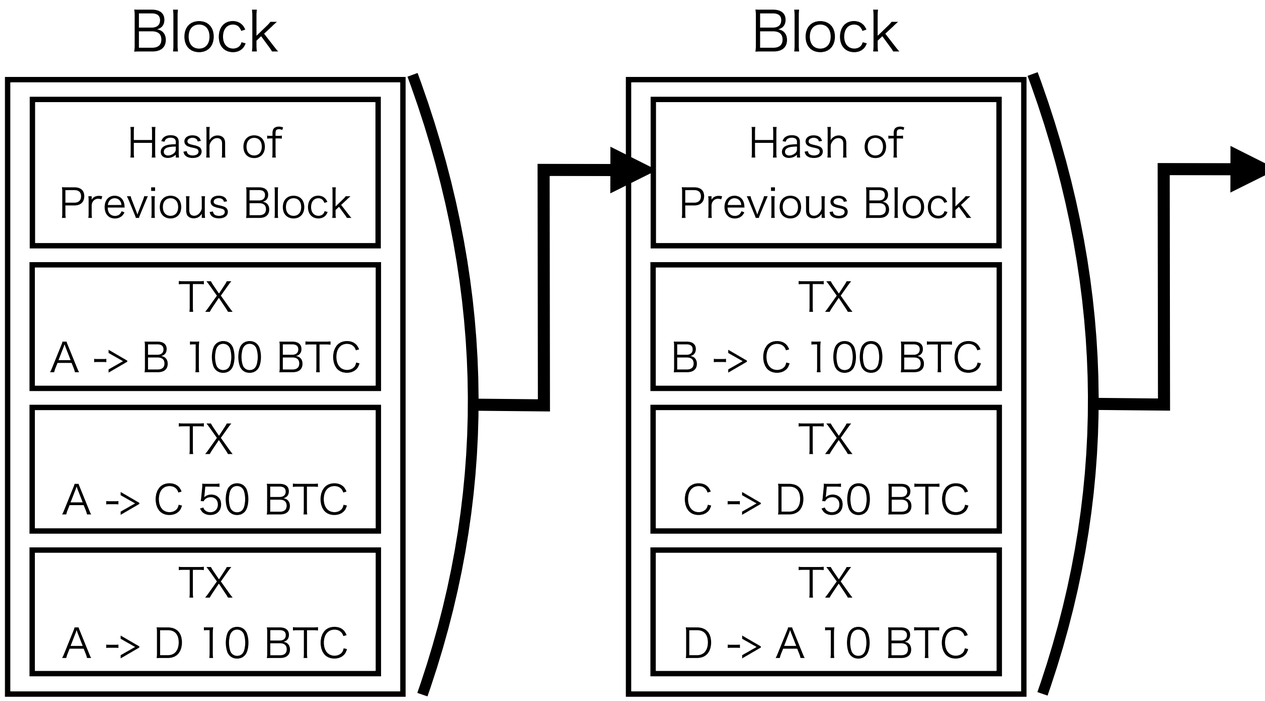


Bitcoin Blockchainサイズの変遷

Data cited from Blockchain.com

TXとBlockのデータ構造

- **Transaction (TX)**
  - 典型的にはTXの正当性は各ノードによって暗号学的に検証可能
  - 新規TXを受信すると、各ノードは検証を行い、通過したものを保存
- **Block**
  - 検証済みTXから「Block」を構成
  - Blockサイズは最大1MB
  - 直前のBlockの暗号学的ハッシュ値を含む
  - ブロックのチェーン構造から、ブロック全体の連なりを「**Blockchain**」と呼ぶ

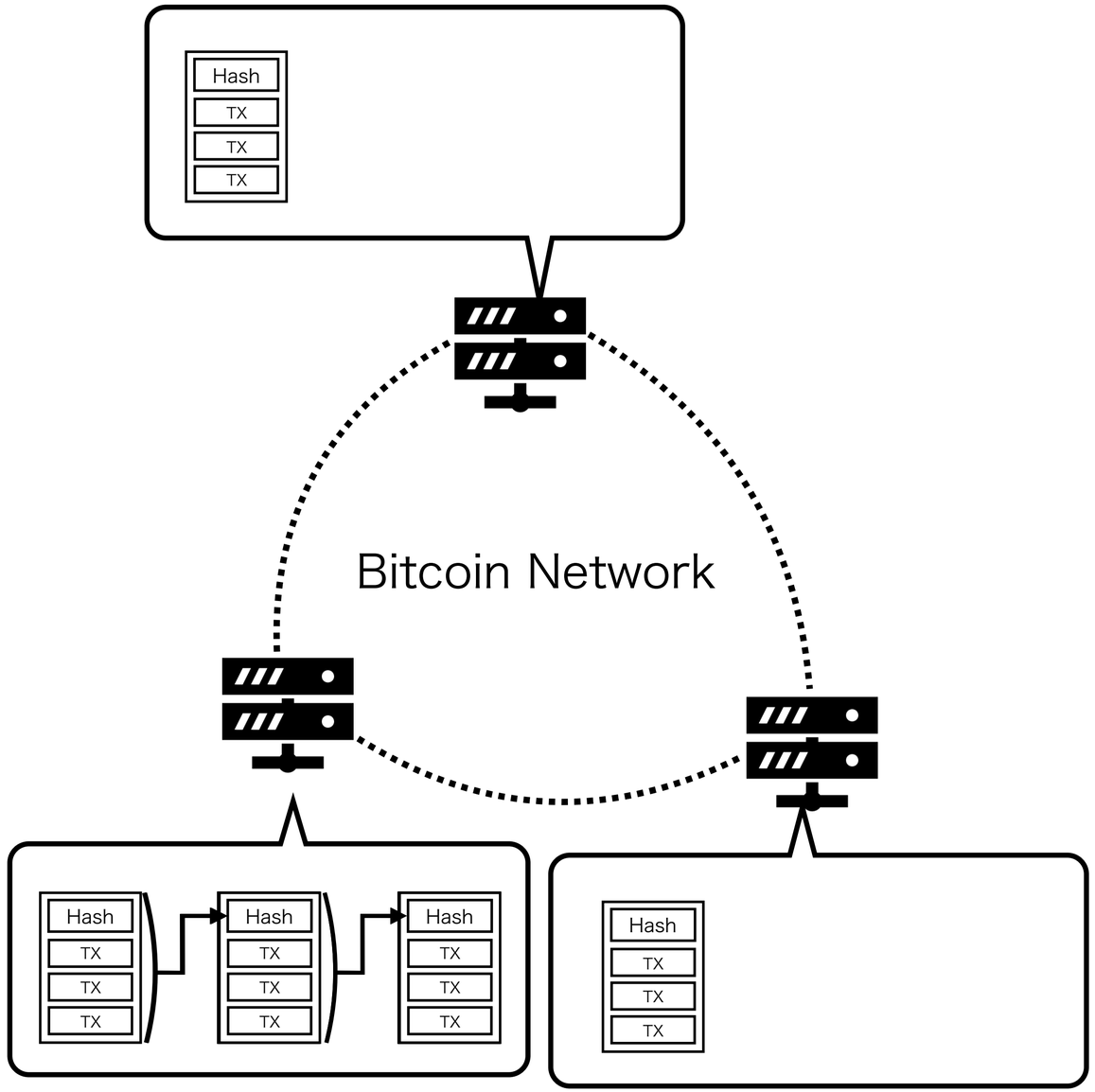


Blockの検証とForkの解決

- **Blockの検証**
  - 新規Blockを受信すると、各ノードは検証を行い、通過したものを保存
  - 検証を行うポイント
    - 含まれるTXは正しいか？
    - 直前のブロックのハッシュ値を含むか？

- **Fork**
  - 矛盾するBlockから一つを選択し解決
  - Blockchain全体にアクセス可能なことが必要

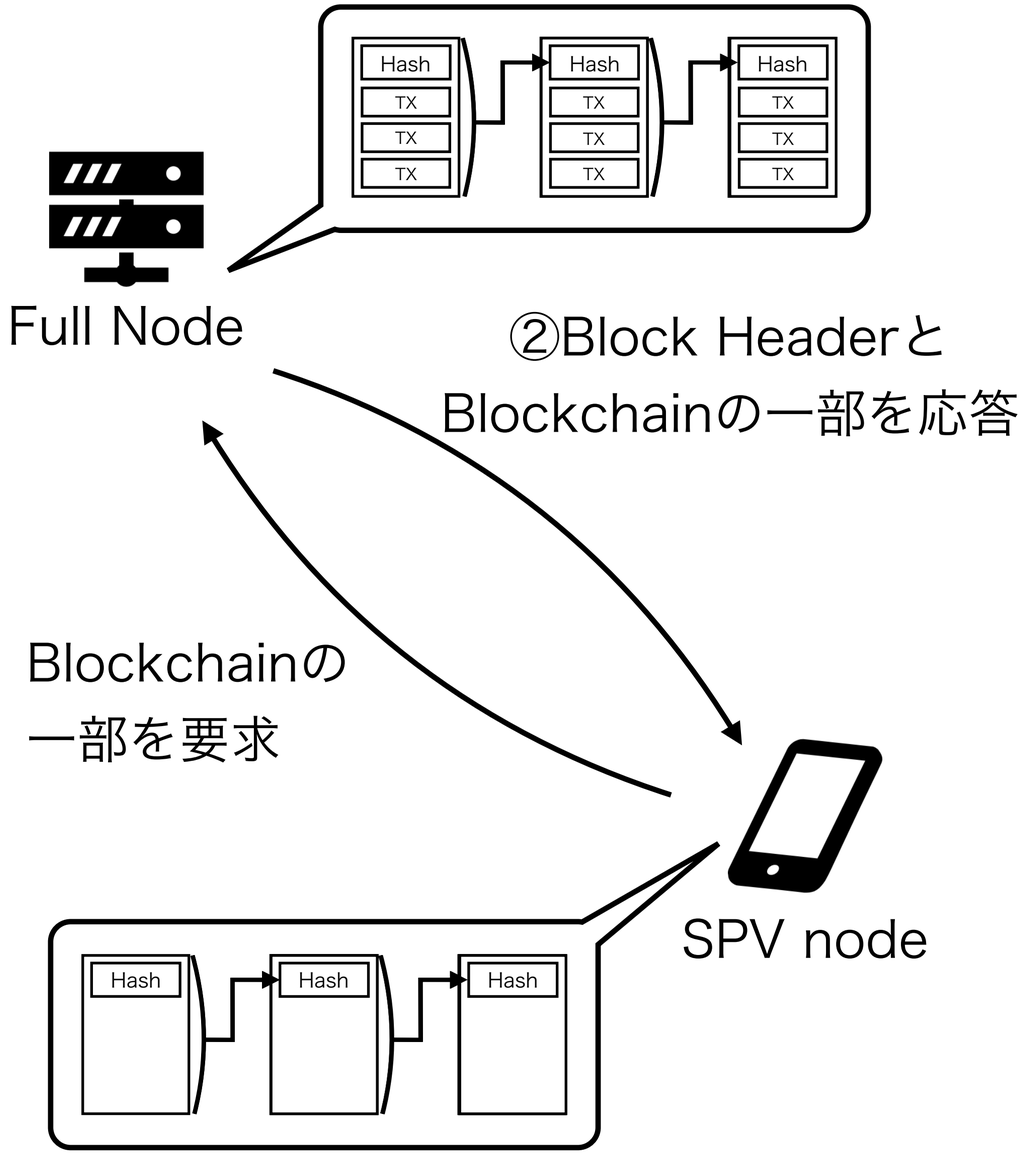
Full NodeはBlockchain全体を保持するため、独立して検証作業を行うことが可能



SPV Node

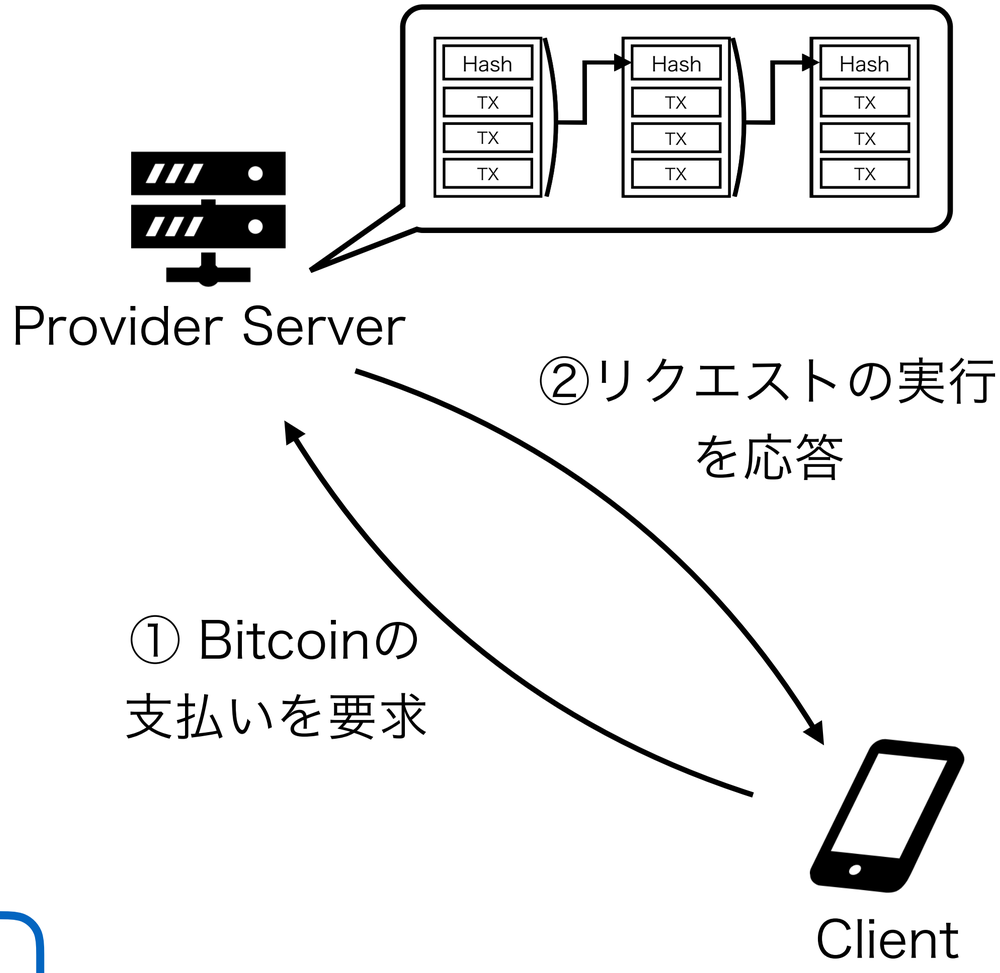
- **Simple Payment Verification (SPV)**
  - Full Nodeを信頼することで、Blockchain全体を持たずともTXとBlockの検証を可能
- **SPVの動作**
  - あるTXがBlockchainに含まれるかどうかを検証
  - Full NodeからBlockの一部(Block header)を取得し検証

SPVノードの可用性は
依存するFull Nodeの可用性に依存



仮想通貨交換所

- **仮想通貨交換所**
  - 暗号通貨と法定通貨の交換
  - 暗号通貨による支払いサービス
- **仮想通貨交換所クライアントアプリ**
  - Webブラウザやスマートフォン上で動作
  - 暗号通貨ノードではない



仮想通貨交換所クライアントアプリの可用性は
サービス提供者のサーバ(Full Node)の可用性に依存

Bitcoinノードの種類と利用

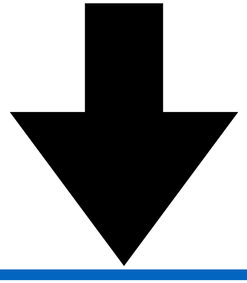
	Full Nodeを運用	SPV nodeを運用	仮想通貨交換所
利点	独立した検証	必要なストレージ 容量の削減	軽量
要件	Blockchainを保持するのに 十分なストレージ容量	Full Nodeを 信頼する必要性	サービス提供者を 信頼する必要性
問題	十分なストレージ容量の見積もりは不可能	独立した検証の欠落	

ストレージ容量の限られたデバイスではBitcoinを動作させることができない

問題と解決策

ストレージリソースの限られたデバイスでの新しいノードスキームの必要性

- 十分なストレージ容量の見積もりは不可能
- 独立した検証の欠落

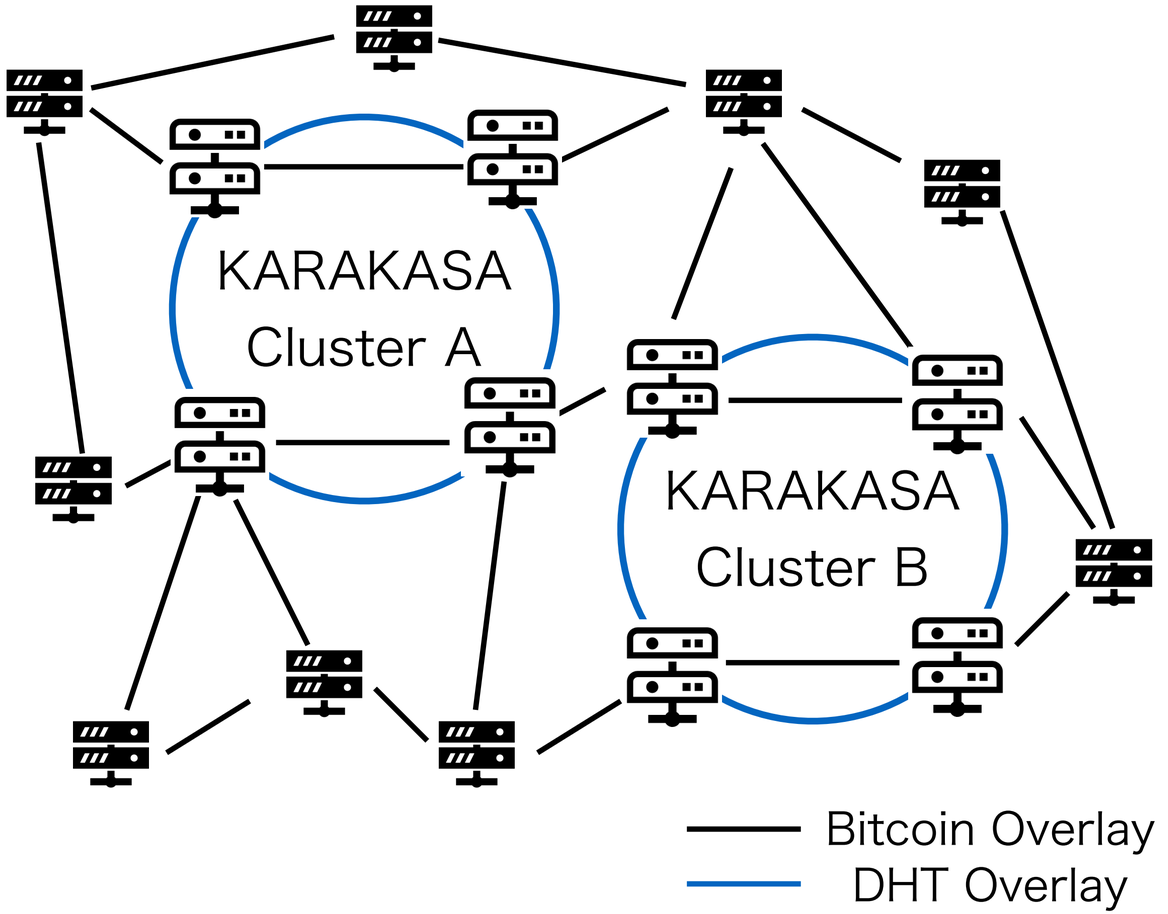


- 1ノードあたりの必要なストレージ容量の削減
- スケール可能なストレージ
- Blockchain全体へのアクセスが可能なことを維持

提案手法: KARAKASA

KARAKASA: DHT(分散ハッシュテーブル)をベースにした分散ストレージを用いたBlockストレージの負荷分散スキーム

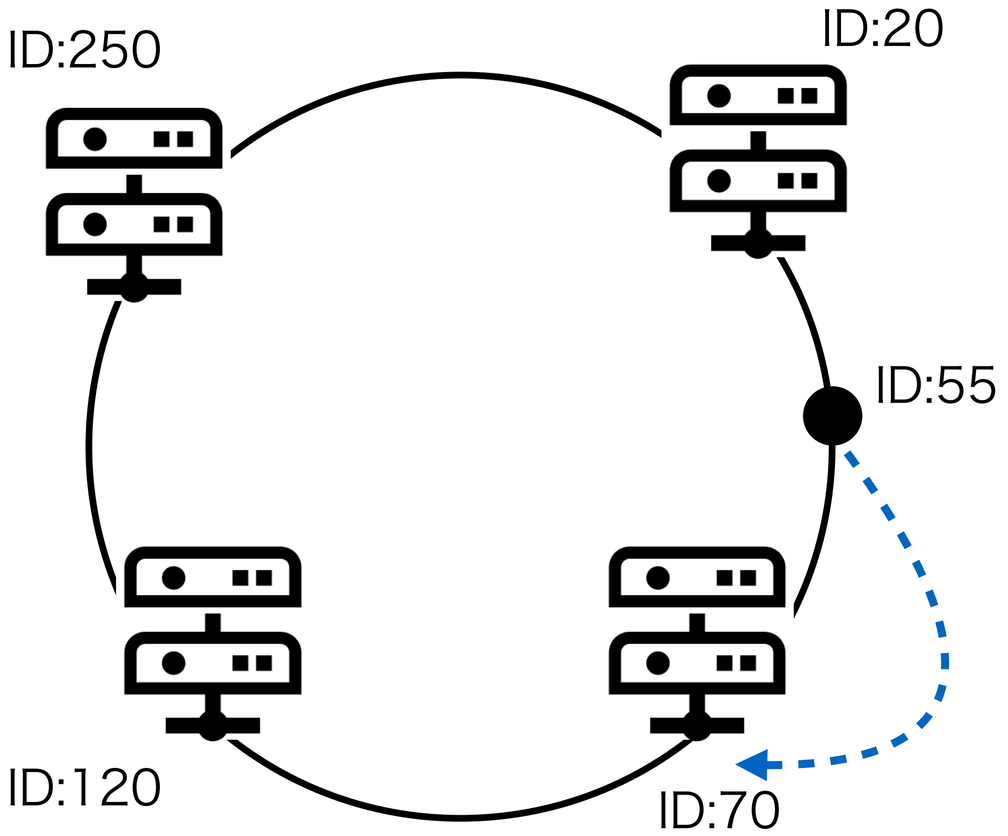
- **DHT**
  - P2Pネットワーク上での効率的なKey-Valueの割り当てと検索を実現
- **ストレージの負荷分散**
  - DHTのアルゴリズムに応じて、各ノードはBlockchainの一部を保持
  - BlockとTXの検証時にDHTクラスタへ読み出しを要求



分散ハッシュテーブル (DHT)

P2Pネットワーク上でハッシュテーブルを共有する仕組み

- **Chord**
  - リング上のオーバーレイネットワークを利用するDHT
  - 各ノードは効率的なルーティングのためのルーティングテーブルを保持
  - NノードでDHTのネットワークが構成される時、1つKey-Valueの読み出しにかかるメッセージ数は$O(logN)$



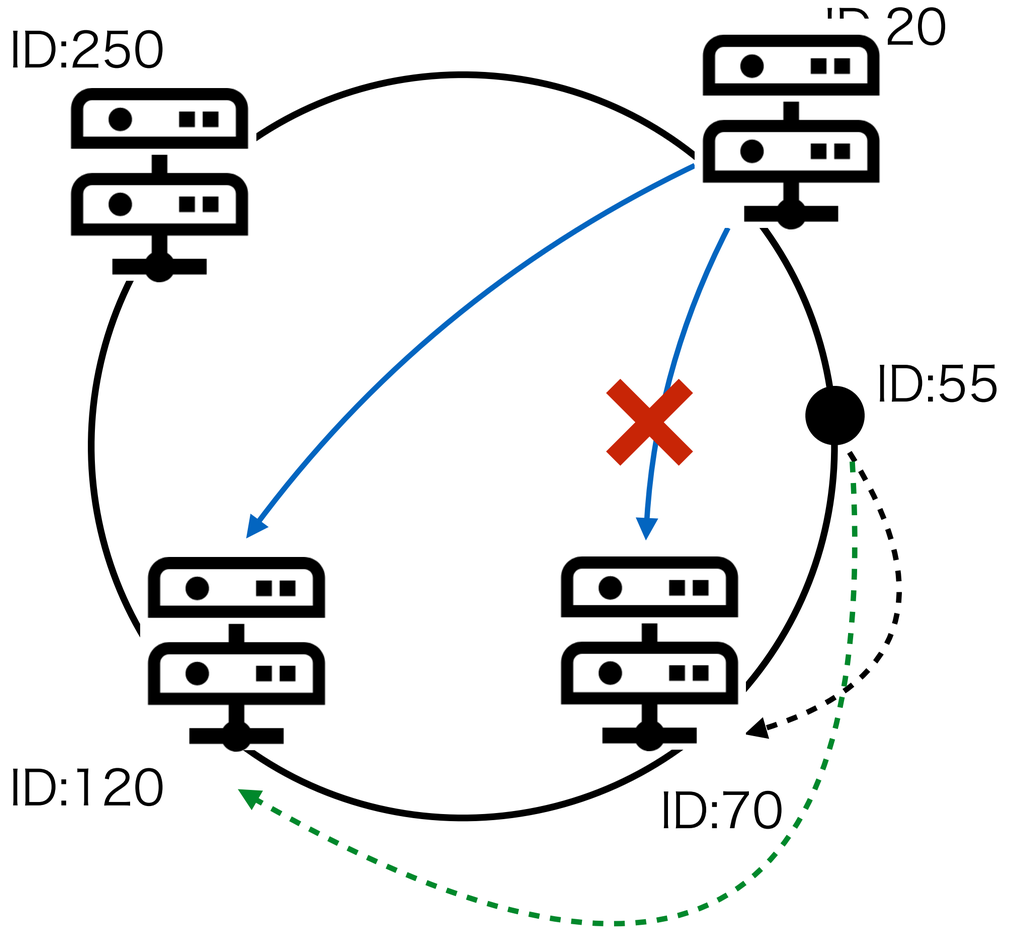
Routing table of node 70

	ID	IP
Predecessor	20	192.168.56.3
Successors1	120	192.168.56.4
Successors2	250	192.168.56.5

DHTのセキュリティ

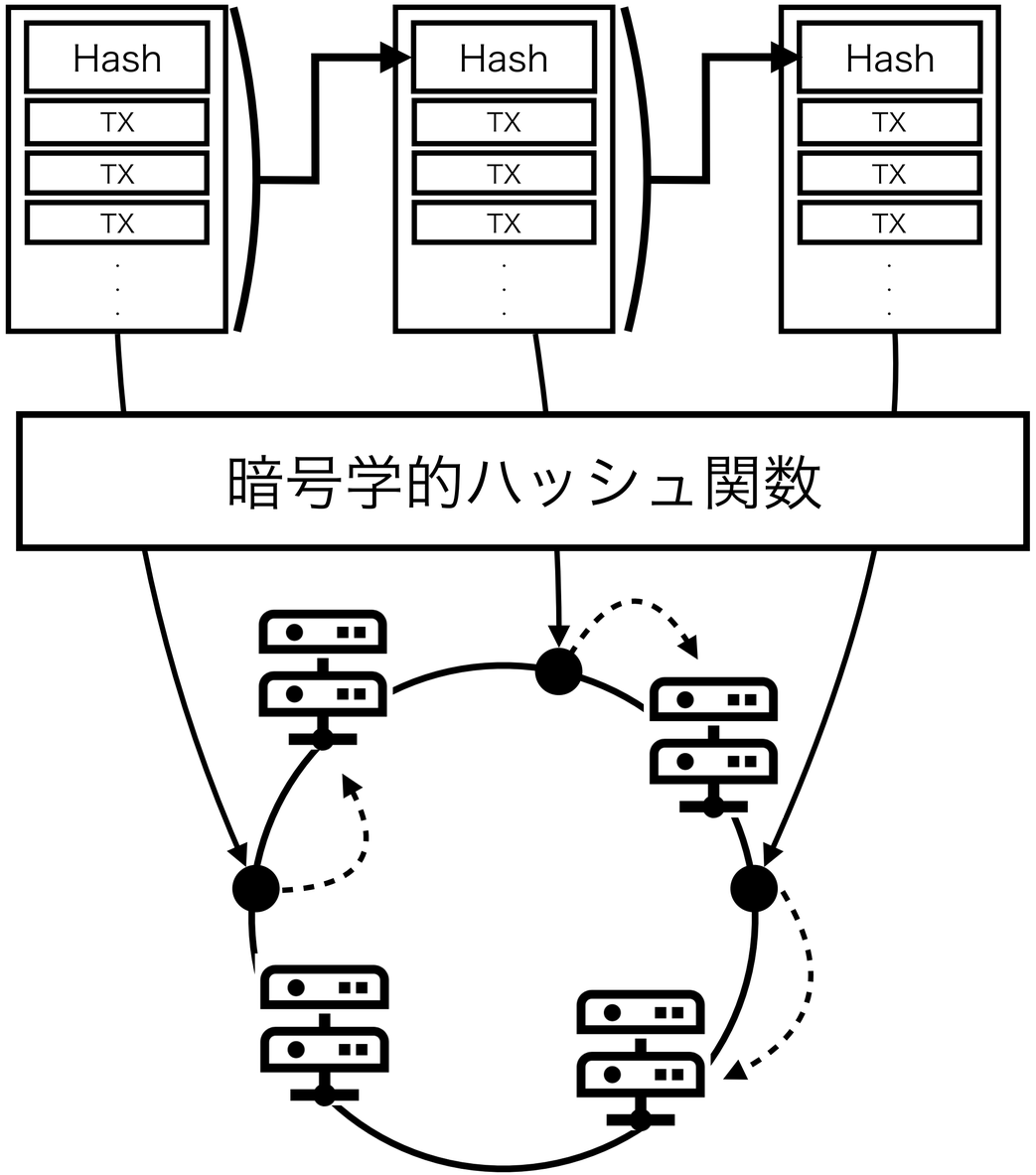
- **DHTへの攻撃**
  - Sybil Attack
  - Eclipse Attack
  - Routing and Storage Attacks
- **Replication（複製）**
  - 対障害耐性を保証
  - e.g. 近接ノードへの複製

DHTのセキュリティスキームは
KARAKASAにおいても利用可能



KARAKASAの動作

- **KARAKASAの初期とノードの参加**
  - 鍵によって参加ノードを認証
- **Blockchainの分散保持**
  - KARAKASAクラスタは「Blockchain」をDHTクラスタ上で動作
- **検証作業**
  - TX:ローカルストレージ内のUTXOsetを利用
  - Block:DHTクラスタへ適宜読み出し



評価概要

KARAKASAスキームを2つの観点から分析し、
Full NodeとSPV nodeと比較

1ノードあたりのストレージ容量

➡ KARAKASAノード1つに要求されるストレージ容量を分析

TXとBlockの独立した検証

➡ Bitcoinシナリオ上でのDHTのセキュリティ分析

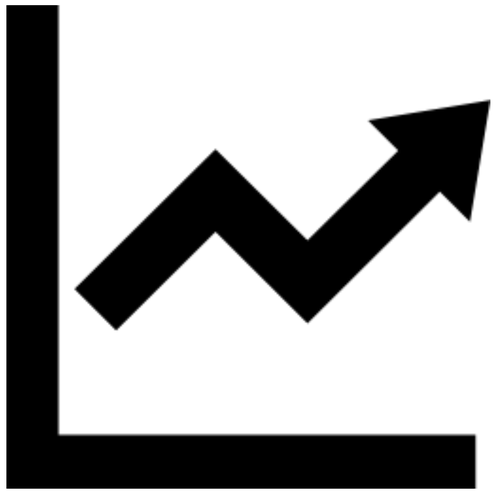
KARAKASAのパフォーマンス分析

手法

- 要求されるストレージ容量の推定
- KARAKASAノードのシミュレーション
  - Overlay Weaver上に実装されたChordを利用

分析ポイント

- 1ノードあたりのストレージ容量
- 分散保持によるメッセージングオーバヘッド



分析パラメータ

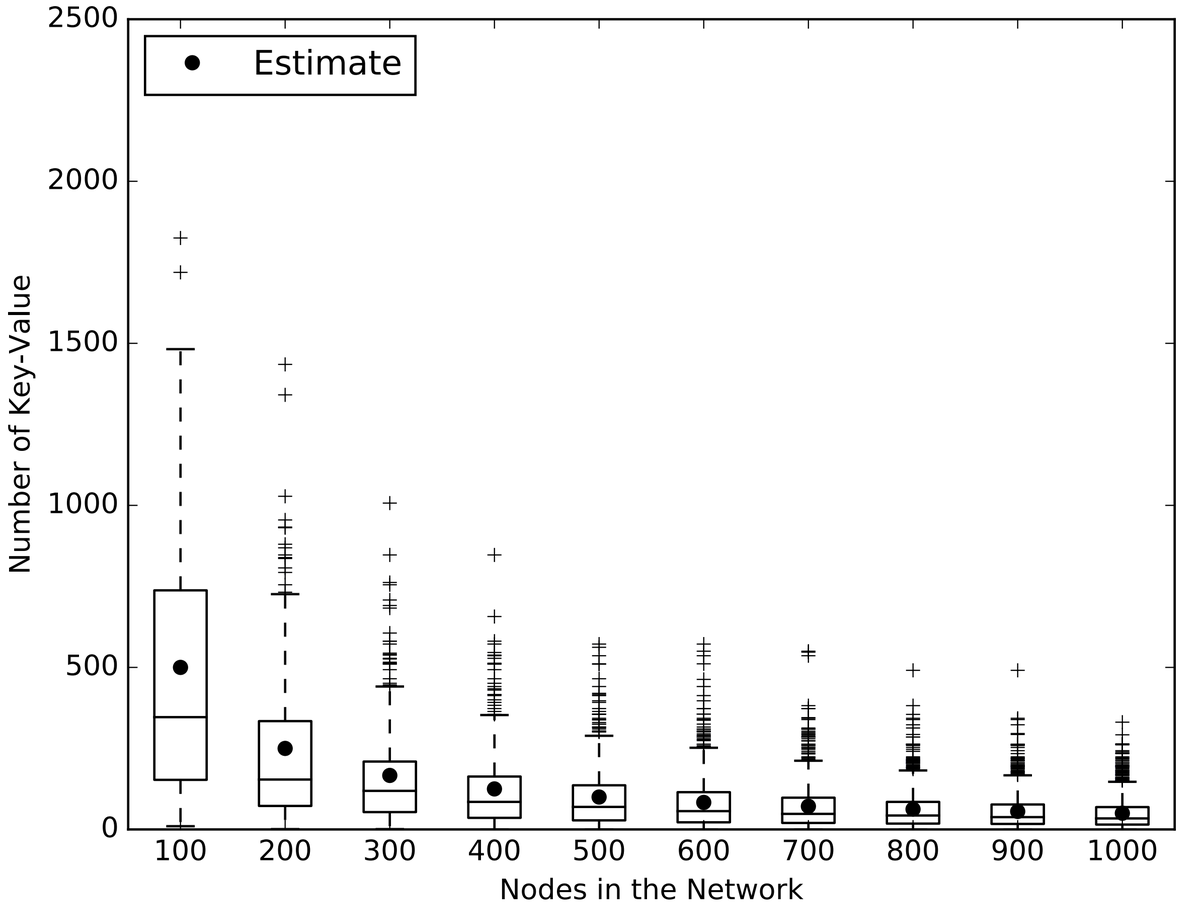
項目	記号	制約	概要
Blockサイズ	$BlockSize$	$BlockSize = 1MB$	Blockのサイズ
Block数	$BlockCount$	$BlockCount \geq 1$	Bitcoinネットワーク上のブロック数
Node数	N	none	KARAKASAクラスタのノード数
Successor数	Suc	$Suc \leq N - 1$	1ノードが持つSuccessorの数
複製数	R	$R \leq Suc$	複製の数

1ノードあたりのストレージ容量

$$StorageSize_{FullNode} \approx BlockCount \cdot BlockSize$$

$$StorageSize_{KARAKASANode} \approx \frac{BlockCount \cdot BlockSize}{N}$$

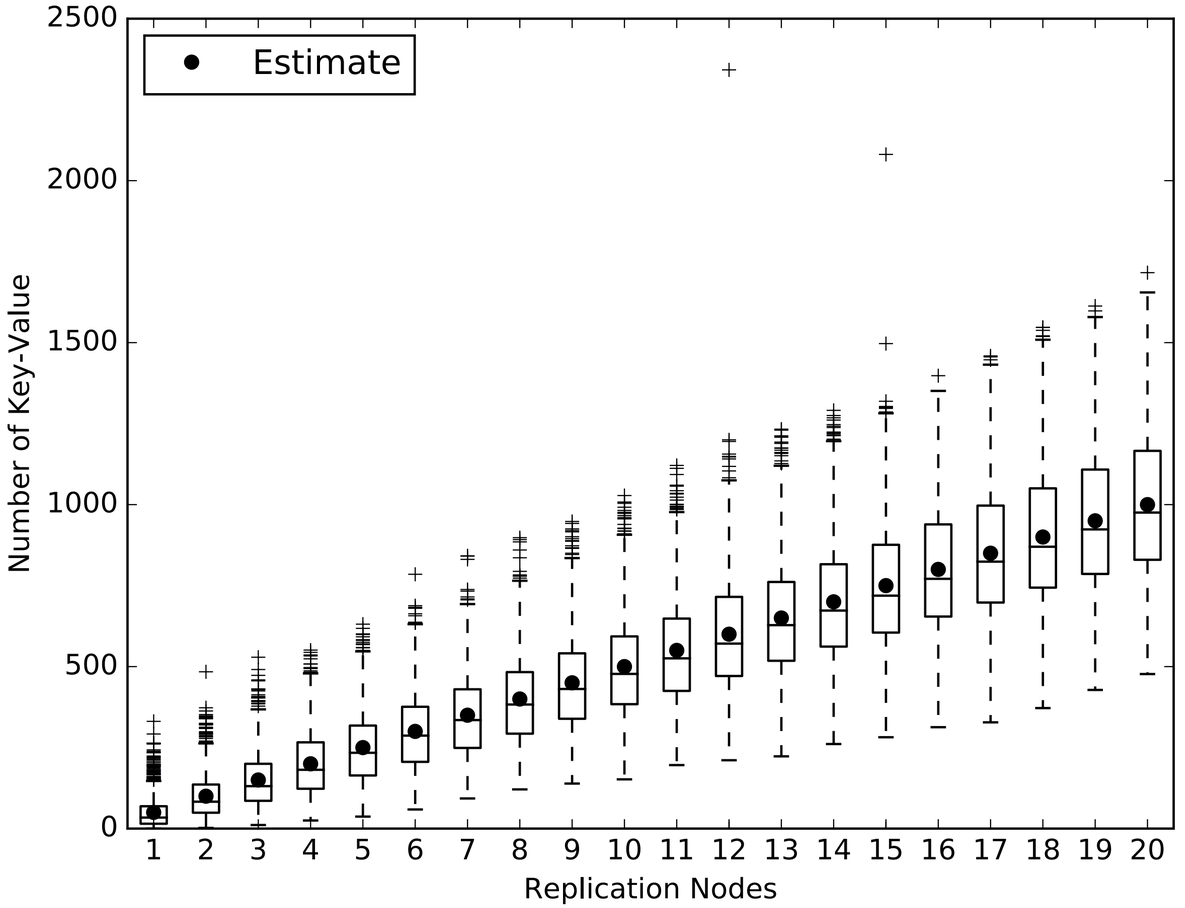
- **シミュレーションシナリオ**
  1. Nノードをエミュレーション
  2. 50000 Key-ValueをDHT上に保存
  3. 各ノードが持つKey-Valueの数を確認



複製を考慮したストレージ容量

$$StorageSize_{KARAKASANodeWithReplication}$$
$$\approx \frac{BlockCount \cdot BlockSize}{N} + \frac{BlockCount \cdot BlockSize}{N} \cdot R$$
$$= \frac{BlockCount \cdot BlockSize}{N} \cdot (R+1)$$

- **シミュレーションシナリオ**
  1. 1000ノードをエミュレーション
  2. 50000Key-Valueを保存
  3. 各ノードが持つKey-Valueの数を確認

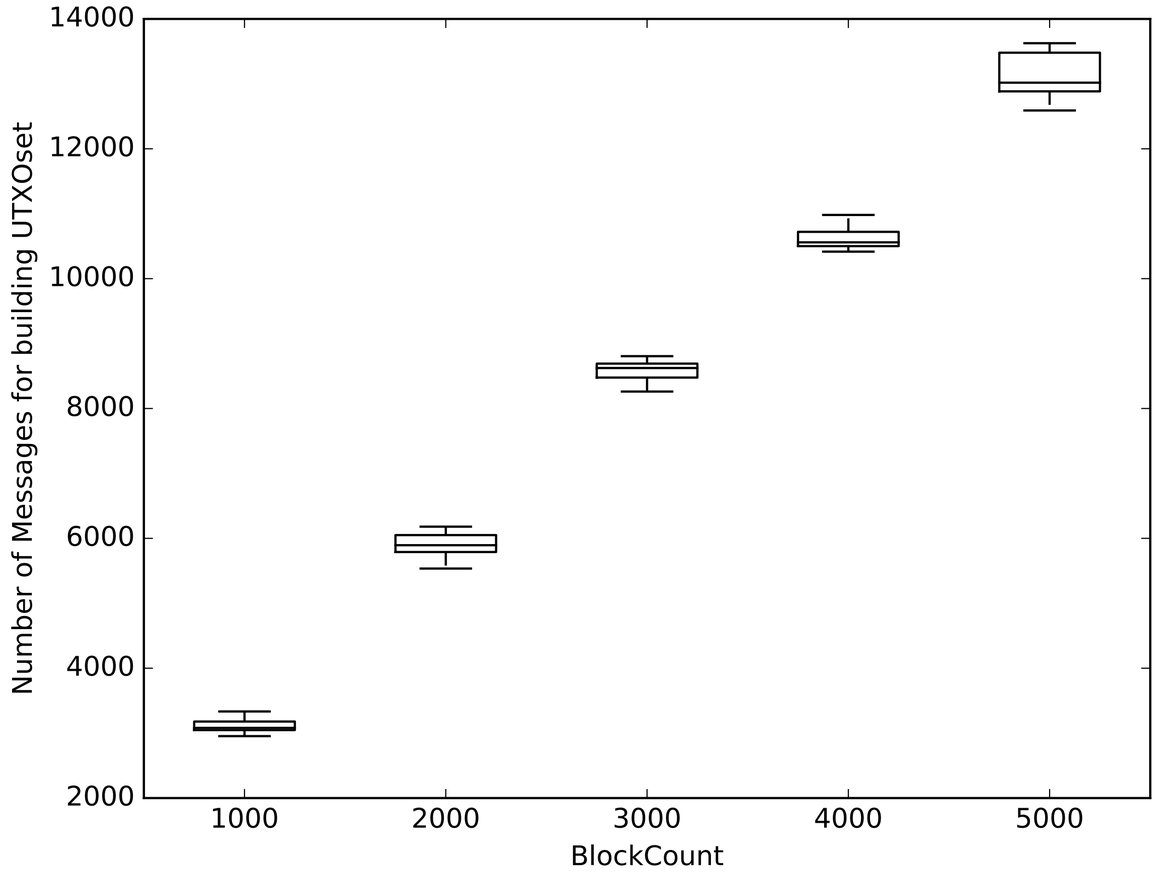


メッセージングオーバーヘッド

$$Overhead for building UTXOset \approx BlockCount * O(logN)$$

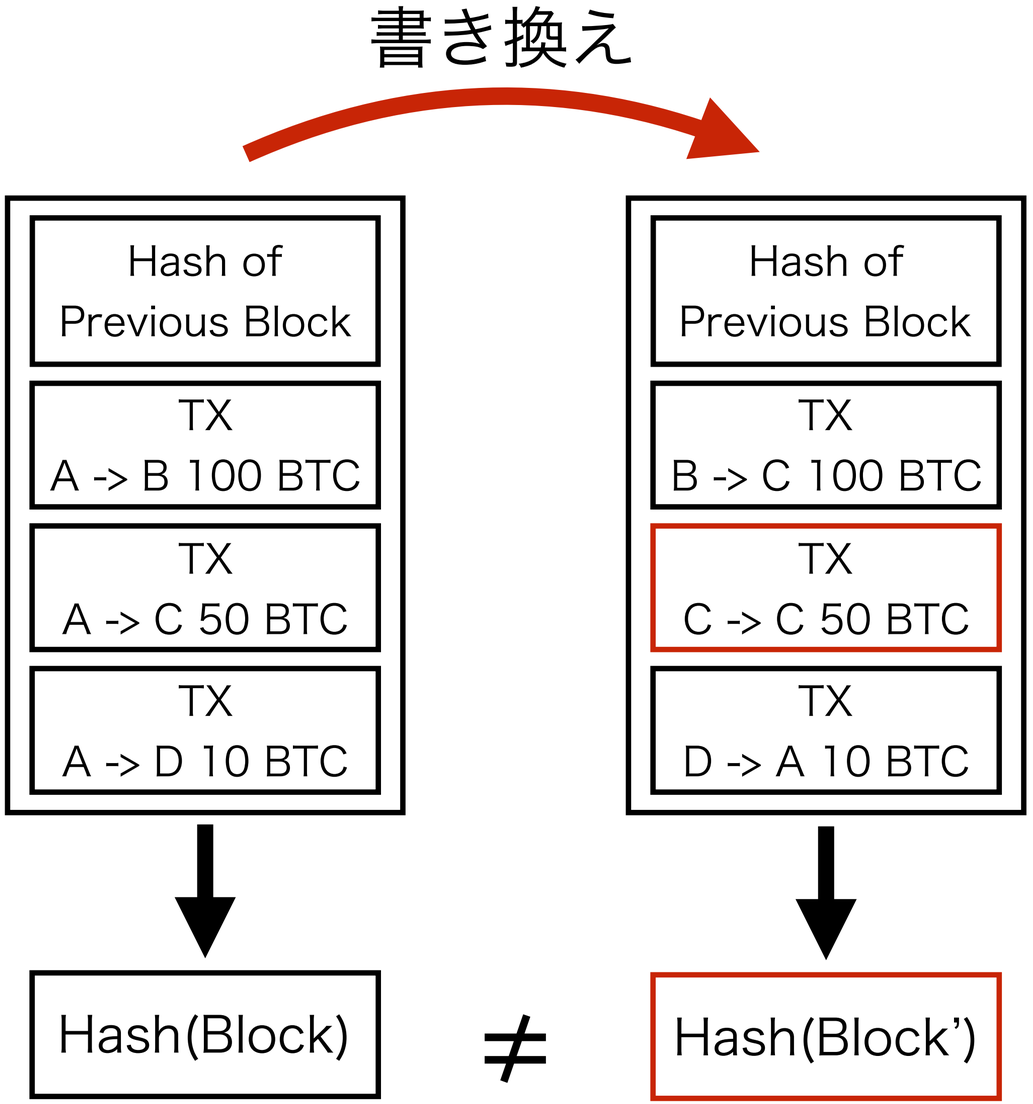
- KARAKASAノードが1Blockを読み出す際のメッセージ数は $O(logN)$
  - UTXOsetを構築する際、ノードはBlockchain内の全てのBlockを読み出す

- **シミュレーションシナリオ**
  1. 1000ノードをエミュレーション
  2. N Key-Valueを保存
  3. 全てのKey-Valueを読み出し
  4. メッセージ数を計測

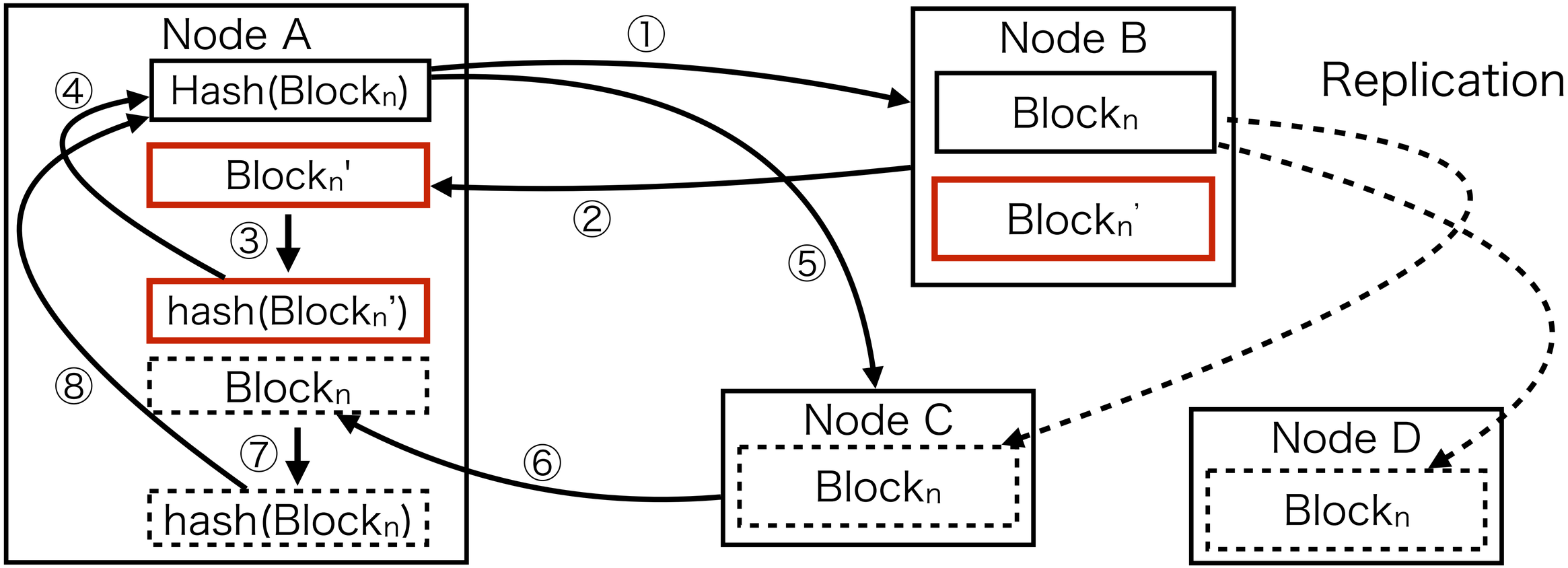


独立した検証の分析

- **独立した検証の可用性**
  - ノードは過去のTXとBlockを正しく読み出せる必要
  - ➡ KARAKASAが独立した検証が可能かどうかはDHTが正常に動作するかどうかに依存
  - ➡ DHTのセキュリティに依存
- **TXの書き換え攻撃**
  - 二重支払いのために、攻撃者は対象のTXを書き換えるか削除することでTXを取り消す



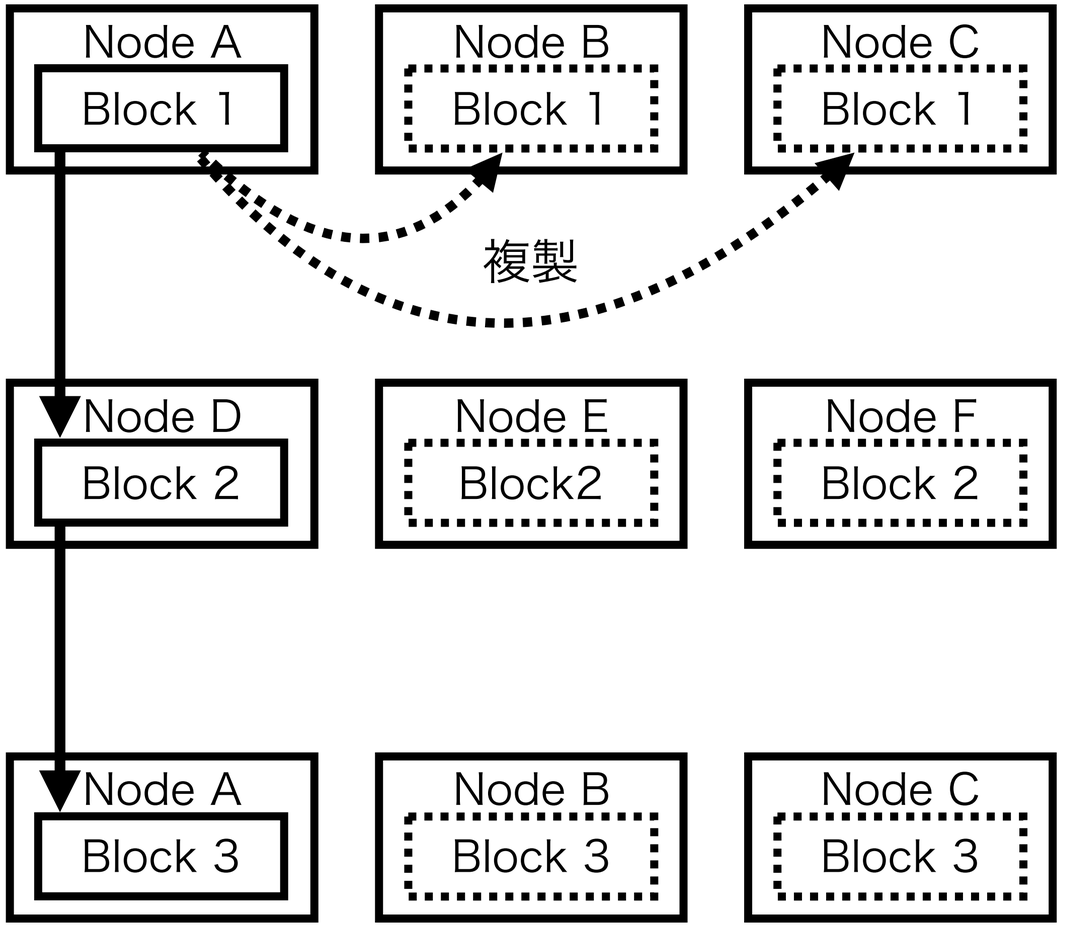
Blockの取得とレスポンスの検証



もし一つのBlockが書き換えられても、書き換えられていないBlockを複製から取得することが可能

TX書き換え攻撃の困難さ

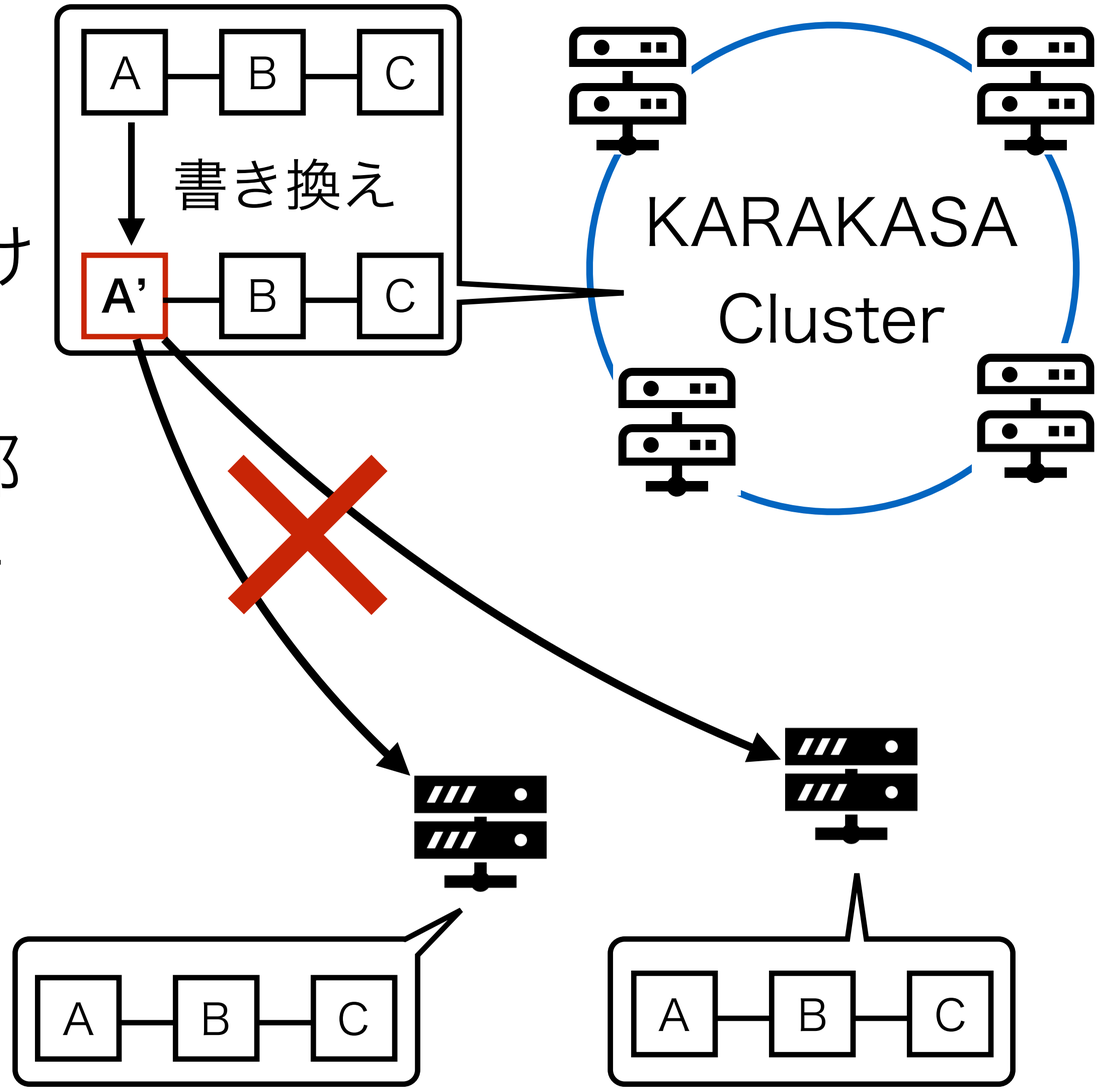
- **困難さの一般化**
  - R 個の複製を作成し、対象のBlock上に $Block$ 個のBlockが続いている時
  - ➡ 攻撃者は $Block \cdot R$ Blockを書き換える必要



ストレージ容量とセキュリテイに
トレードオフの関係性がある

TX書き換え攻撃の影響

- **他のノードへの影響**
  - クラスタ内での書き換えが成功しても、Bitcoinネットワーク内の数ノードが受け入れたに過ぎない
  - TXを取り消すには、KARAKASAの外部でも受け入れられるように書き換えを行わなければならない



KARAKASA内部でTXの書き換えをするだけでは、TXの取り消しにはならない

Full NodeとSPV nodeとの比較

ストレージ容量

	概要	比較
Full Node	· Blockchain全体	Large
KARAKASA node	· Blockchainの一部 · クラスタのノード数増加によってスケール可能	Middle
SPV node	· Blockの一部	Small

独立した検証作業

	概要	独立しているかどうか？
Full Node	· 独立し、ローカルで実行可能	独立
KARAKASA node	· TXにおいては独立 · Forkの解決はクラスタ内の複製数に依存	独立
SPV node	· Full Nodeに依存	依存

結論

提案概要

- 新しいBlockchainストレージ負荷分散スキーム「KARAKASA」の提案
- DHTクラスタが分散的にBlockchainを保持
- 1ノードに必要とされるストレージ容量の削減
- 複製によって他の特定のノードを信頼することなく動作可能

- **貢献**
  - ストレージ容量の限られたデバイスでの新たなノードタイプの選択肢
  - KARAKASAノードを信頼のエンドポイントとすることが可能
- **Future Works**
  - メッセージングオーバーヘッド
  - 複製手法の最適化
  - KARAKASAを取り入れたBitcoinエコシステム