4ZD-04

ハッシュグラフ技術を用いたスケーラブルな価値交換システムの構築とその性能評価

三島潤平 †　相田仁 †

東京大学大学院工学系研究科電気系工学専攻 †

1　はじめに

金融や流通などの分野で，ブロックチェーンをはじめとする分散台帳技術（DLT）を活用した価値記録・交換システムの導入が検討されている．ハッシュグラフ [1] は，改ざん耐性を備える DLT の中でもマシンパワーに寄らない公平な合意形成と単位時間あたりの取引処理性能に優れた，有向グラフ（DAG）型の DLT として知られている．しかし，ハッシュグラフの合意形成に参加する端末数が増加すると，取引の発行から処理順序決定までの時間が長くなってしまい，取引完了までの待ち時間の面で他の DLT に対する優位性を失ってしまうという問題がある．そこで本研究では，ハッシュグラフの合意形成参加端末数が増えても取引の遂行までの時間に対するスケーラビリティを確保できるようにするために分割ハッシュグラフを提案する．分割ハッシュグラフでは端末をグループ分けし，グループ内とグループ間に関する小規模なハッシュグラフでより高速な合意形成を行う．本稿では，分割ハッシュグラフの概要・シャード間取引のプロトコル・シミュレーションによる分割ハッシュグラフの性能評価について報告する．

2　関連研究

2.1　ブロックチェーン技術におけるシャーディング

Zilliqa [2] は，シャーディングを用いたブロックチェーンプラットフォームの 1 つである．Zilliqa では，ノードをシャードと呼ばれるグループに分け，それぞれのシャード内で PBFT やシュノア署名を活用した高速な合意形成を可能にしている．シャードごとに合意形成の正当性を確保するために必要なノード数を保ちながらも，各シャードで並行して合意形成を行うため，単位時間あたりに処理できる取引数が多い．本研究では，ハッシュグラフにシャーディングを適用することでハッシュグラフのサイズを縮小し，最終的な取引の遂行までの時間を短くすることを目指す．

2.2　DAG 型のコンセンサスアルゴリズムの性能改善

Jointgraph [3] は，ハッシュグラフのスケーラビリティについてノードの参入容易性と承認時間の短縮の両面から解決を試みた．Jointgraph ではスーパーバイザノードを設定し，スーパーバイザノードが作成したイベントを基準にイベントの処理順序決定や DAG 構造のスナップショット作成を行う．ハッシュグラフに比べて，スーパーバイザノードを信頼することでイベントの処理順序決定に必要なコミュニケーション数を減らすことができ，ハッシュグラフに比べてスループットや取引承認の遅延を約 3 倍改善している．Jointgraph はコンソーシアム型を想定した設計で，ノード数が膨大となったときのスケーラビリティには必ずしも対応できていない．本研究では，ノード数が膨大となっても取引の遂行までの時間が長くなりすぎないことに焦点を当てた提案を行う．

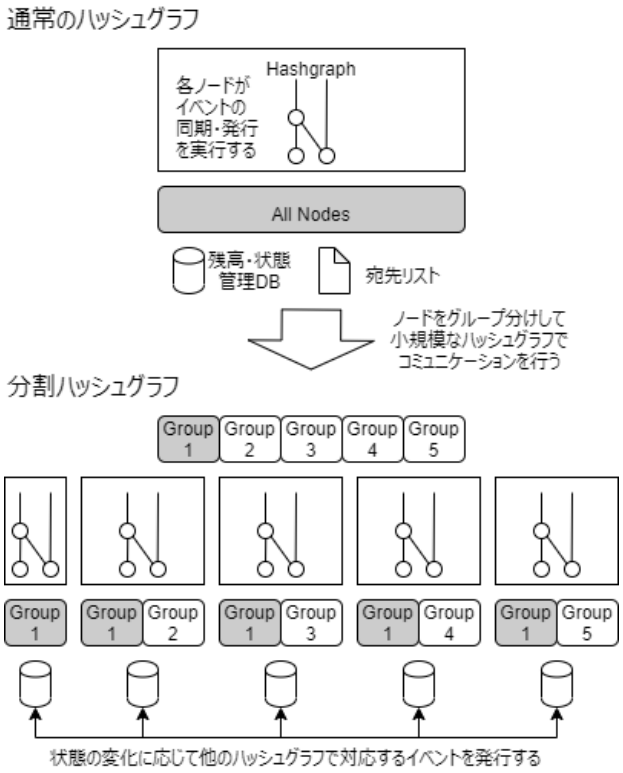


図1: 分割ハッシュグラフの概念図.

3　分割ハッシュグラフの提案

3.1　分割ハッシュグラフの概要

図 1に分割ハッシュグラフの概念図を示す．分割ハッシュグラフは，ノードをいくつかのグループに分け，自身のグループ内と自身と他のグループ間のメンバーをそれぞれ合意形成のメンバーとする独立した複数のハッシュグラフと状態管理データベースを持つ．通常のハッシュグラフと同様に，各ハッシュグラフで合意形成・処理順序決定されたイベントの内容に従って対応するデータベースの状態を更新する．別のシャードのメンバーとの取引を可能とするプロトコルが必要だが，1 つ 1 つのハ

Construction and Evaluation of Scalable Exchange System Using Hashgraph

Junpei MISHIMA† and Hitoshi AIDA†

† Aida Laboratory,

The University of Tokyo

7-3-1 Hongo, Bunkyo-ku, Tokyo, Japan

{junpei, aida}@aida.t.u-tokyo.ac.jp

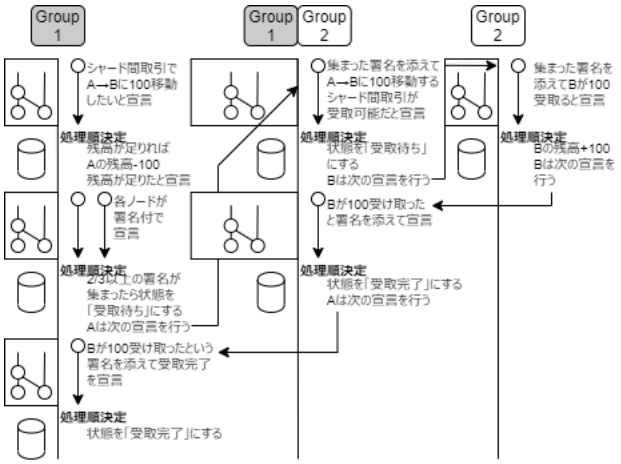


図2: シャード間取引の流れ．グループ 1 に属する A からグループ 2 に属する B に対して残高 100 の移動を行う場合の例．最初の処理順決定の段階で A の残高が充足していれば次に進むが，残高が不足している場合は通常の単一ハッシュグラフでの処理と同様に何も状態を変化させずに終了する．イベントのハッシュ値と取引内容に対する署名を利用して異なるハッシュグラフ間での情報の整合性を取る．

ッシュグラフの規模が小さいため，データベースの更新までの時間が通常のハッシュグラフよりも短縮され，取引の遂行までの時間も短くなることが期待できる．

3.2 シャード間取引のプロトコル

図 2にシャード間取引のうち，あるグループから他のグループのメンバーに残高を移動する取引のプロトコルを示す．自シャード内で残高の確認・仮引き落としを宣言するイベントを発行して承認・処理したのち，取引に関連する 2 者にまたがるシャードで残高の受け取りが可能であることを宣言する．宣言は取引の当事者が行うなど，各段階によって定めており，一意に状態が遷移する．

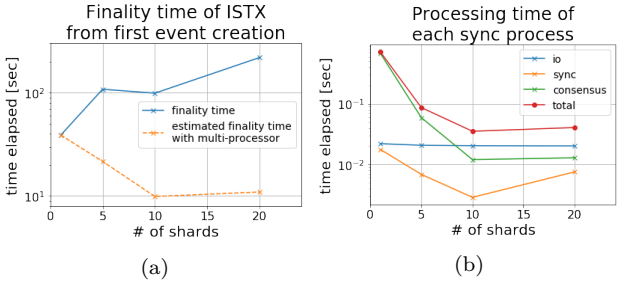
4 分割ハッシュグラフの性能評価

4.1 実験条件

100 ノードが参加するハッシュグラフにおいて，分割無しの場合と分割数 5・10・20 の場合とで自シャードで 10,000 イベントの処理順序が決定し，処理が完了するまでコンセンサスアルゴリズムを動作させ，取引の遂行までに要する時間と各種処理に要する時間をシミュレーションによって測定した．マシンは MacBook Pro（13 インチ，2017 年モデル）を使用し，CPU は Intel Core i5 2.3 GHz，メモリは8 GBであった．ノード間の通信遅延は片道10 ms，通信速度は100 Mbpsとし，各ノードはできるだけ短い間隔で通信と状態の更新を行うこととした．ハッシュグラフのランダム性を再現するため，ハッシュグラフ同期のための通信を開始するノードをランダムに選択するようにし，選択されたノードは10 %の確率で新たな取引を発行し，イベントに書き込むようにした．

4.2 実験結果

図 3に分割ハッシュグラフの性能評価結果を示す．図 3(a) を見ると，シャード間取引の遂行までの時間について分割無しの場合が最も短くなる．これは，分割無しのハッシュグラフでは全てのノードに対して残高の移動を 1 度のイベント発行と承認で完了させることができるためである．本シミュレーションは，各ノードがシングルスレッドで各シャードの処理を順番に行うようにしたため，複数 CPU で並行処理を行った場合，分割ハッ



(a) (b)

図3: 100 ノードによる分割ハッシュグラフの性能評価結果．(a) シャード間取引（ISTX）の最初のイベント発行から，自シャードで状態が「受取完了」になるまでの時間の平均．(b) 各ノードが同期処理を行う際のネットワーク通信（io），同期準備（sync），イベント処理順序決定（consensus）の各段階の所要時間とその合計（total）の平均．

シュグラフの性能は向上すると考えられる．高性能なエッジサーバーとして 96 コアの CPU を使うと想定すると，今回の実験の全シャードのハッシュグラフは並行処理が可能であり，取引完了時間の予測値はシャード数で除算した値となり，図 3(a) の点線のようになる．この場合は分割数 10，つまり 1 シャードのノード数が 10 ノードの場合に最も取引完了までの時間が短くなる．分割無しの場合は，マルチプロセスで処理を実行しても，同期処理におけるデータの伝播に要する時間を短縮することはできないため，取引完了までの時間は変化しないと考えられる．図 3(b) を見ると，シャード分割数を増やしたときに，イベント処理順序決定の時間が分割無しのときと比べて突出しなくなることが分かる．また，分割数 20 の場合は分割数 10 の場合と比べて同期処理における処理時間の減少が見られないため，シャード分割数の分だけ処理すべきシャード全体で見たときの処理量が多く，その分，分割数 10 のときよりも取引完了までの時間が長くなっていると考えられる．

5 おわりに

本稿では，分割ハッシュグラフを提案し，性能評価を行った．その結果，シャード間取引は 6 段階のコンセンサスを経る必要があり，100 ノードの場合，シングルスレッドで動作する限りにおいて分割無しのハッシュグラフを上回ることができなかった．しかし，小規模に分割した各ハッシュグラフ内での処理時間は分割無しの場合と比べて大きく減少させることができており，マルチプロセスにより各ノードが擁する複数のハッシュグラフの更新を並行して実行するようにすれば，分割無しの場合に比べて，シャード間取引のオーバーヘッドを上回る効率を出すことができる可能性があることが分かった．今後，より多くのノードが参加する状況下で，マルチプロセスによる並行処理も加味したより詳細な実験により，分割ハッシュグラフにおいて取引遂行までの時間を最適化するための条件について検討が必要である．

参考文献

[1] L Baird. Hashgraph consensus: fair, fast, byzantine fault tolerance. Technical report, 2016.
[2] The Zilliqa team. The Zilliqa Technical Whitepaper. https://docs.zilliqa.com/whitepaper.pdf, (Accessed on 2020-01-10), 2017.
[3] X Fu, H Wang,P Shi, X Ouyang, and X Zhang. Jointgraph: A DAG-based efficient consensus algorithm for consortium blockchains. Journal of *Software: Practice and Experience*, pp.1–13, 2019