ブロックチェーンネットワークに対する Plumtree アルゴリズムの適用に関する一検討

A study of Applying Plumtree Algorithm for Blockchain Networks

北川　雄介†　　首藤　一幸‡　　水野　修†　　坂野　遼平†‡
Yusuke Kitagawa　Kazuyuki Shudo　Osamu Mizuno　Ryohei Banno

†工学院大学　　‡東京工業大学
Kogakuin University　Tokyo Institute of Technology

1．はじめに

暗号通貨[1]を支える技術としてブロックチェーンが注目を集めている．ブロックチェーンは，取引などを記録するための台帳をネットワーク上の複数のコンピュータで共有し，内部及び外部からの改ざんを防ぐ仕組みである．しかし，ブロックチェーンを活用していく上で大きな問題となるのが，通信リソースの過剰な消費である．ブロックチェーンは，多数のノードがランダムに相互接続を行い，P2P (Peer to Peer) ネットワークを形成して情報の送信を行う．全ノードに情報を行き渡らせるためにノードが隣接するノードに対して情報を送信する．情報を受信したノードもその情報を別のノードへ送信することを繰り返すことにより，情報を行き渡らせている．そのため，既に受信した情報が別ノード経由で複数回届くため，通信リソースが消費してしまう．そこで，本研究ではランダムなトポロジを持つブロックチェーンネットワークにおいて，Plumtree[2]を適用することで伝播経路を緩やかに固定し，重複した情報を削減する手法を提案する．

2．提案手法

既存のブロックチェーン技術には通信リソースの消費という課題がある．例として Bitcoin のブロックチェーンは，ノード同士でブロックを共有する際に先立って，inv メッセージを送る．inv メッセージを受け取ったノードは，足りないブロックがあるか確認を行う．もし，不足しているブロックがあれば，inv メッセージの送信元ノードに，ブロックを送るというリクエストを getdata メッセージを使って出す．この仕組みにより，ブロックを送るにあたり，相手側のノードがブロックを既に知っているかどうかに関わらず，常にブロックをネットワーク全体に送信することが無くなるため，余分な送信を無くしている[3]．しかし既に受信した inv メッセージが別のノード経由で複数回届くため，通信リソースが消費してしまう．

本研究では，Bitcoin ネットワークをベースに，Plumtree の冗長なメッセージ送信を削減する機能を用いて通信リソースの消費問題を解決する．Plumtree は，効率的にメッセージの配送を行える特徴がある．これはランダムなネットワークにスパニングツリーを構築し，ツリーでのみメッセージが転送されるようにすることで，冗長な送信を減らし，効率化しているためである．Plumtree の概要を，図 1 に示す．提案手法では，まず，Bitcoin と同じ仕組みでブロックをブロードキャストする．その際に，Plumtree のアルゴリズムを用いて，スパニングツリーを構築する．それ以降のブロードキャストでは，構築したツリーを利用する．その際，inv・getdata メッセージのやり取りは省き，ツリーの子ノードに直接ブロックを送る．ノードの増減やネットワーク障害に伴って必要となるツリー修復については、Plumtree のアルゴリズムに従う．

表 1 ブロックサイズ・ノード数

ブロックチェーン	ブロックサイズ	ノード数
Bitcoin	803.565 Kbyte	8370
Bitcoin SV	1.5427 Mbyte	288
Bitcoin Cash	87.685 Kbyte	1281
Dogecoin	15.863 Kbyte	397
Dash	30.295 Kbyte	6165
Ethereum	40.676 Kbyte	6647
Litecoin	28.689 Kbyte	1349
Zcash	6.501 Kbyte	140

表 2 実験パラメータ

パラメータ	内容
NUM_OF_NODES	8370
BLOCK_SIZE	803.565 Kbyte
CBR_USAGE_RATE	0
CHURN_NODE_RATE	0

表 3 実験に用いた PC の諸元

項目	内容
CPU	Intel Core i5-10400 CPU @2.90GHz
RAM	32GB
OS	Windows 10 Pro Education
Java Version	15
SimBlock Version	Version 0.8.0

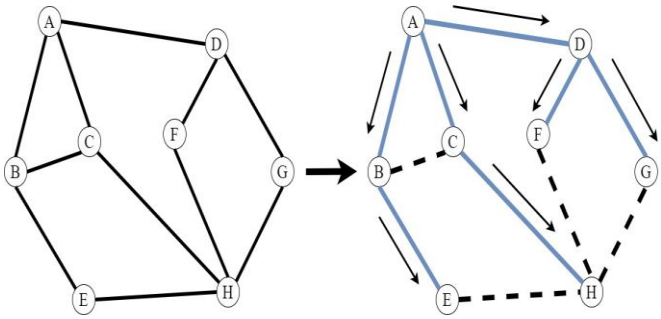

図 1 Plumtree 概要図

3. 評価

ブロックチェーンの情報伝播を模擬できる SimBlock[4]を基に，Plumtree を用いて機能の実装を行う．

この実験は，SimBlock を用いたシミュレーション実験である．SimBlock は Bitcoin の仕組みをベースに実装している．提案手法によるメッセージ数を測定及び，伝播遅延への影響を確認する．測定環境としては，各ブロックチェーンのチャート情報より，各ブロックチェーンに対するノード数及びブロックサイズを参照し測定を行った．ブロックサイズは，2020 年 12 月 15 日時点での平均ブロックサイズを使用し，ノード数も同様に，2020 年 12 月 15 日時点の合計ノード数を使用した．表 1 に参照したデータを示す[5][6][7]．本実験で使用した SimBlock の実験パラメータを表 2 に示す．ノードの増減は無しとし，CBR (Compact Block Realy) は利用しない．それ以外は SimBlock のデフォルト値を利用する．PC の実験環境を表 3 に示す．

メッセージの評価方法としては，各ブロックチェーンに対するノード数及びブロックサイズを参照し，ブロックの伝播を 4 ブロックまでシミュレートした．試行回数は，各ブロックチェーンともに 10 回ずつであり，inv メッセージと getdata メッセージの数をカウントするプログラムを実装して，全ノードの受信メッセージ数の合計を算出した．次に伝播遅延の影響を確認するために，各ブロックの平均遅延を，SimBlock の標準出力で表示される，ブロックが生成されてから当該ノード ID のノードに到達するまでの時間を用いて平均値を求めた．またホップ数は，構築したツリー上のブロック生成元から転送回数を算出するプログラムを実装して求めた．

本稿では Bitcoin を対象として測定を行った．ブロックチェーンに Plumtree を適用した場合のメッセージ数の評価を行った．図 2 に提案手法と従来手法のメッセージ数を示す．縦軸は，全ノードの受信メッセージ数の合計，横軸は測定項目である．提案手法と既存手法を比べるとメッセージ数の削減が行えていることがわかる．この提案手法のメッセージ数は，最初のブロックのやり取りにかかったメッセージのみである．そのため既存の手法よりも余分なメッセージを省けている．次に 図 3 にホップ数と遅延の関係を示す．図 3 では平均ホップ数が多いほど平均遅延時間も上がっている．ホップ数が多くなる理由としては，Plumtree により最初のブロックを送るノードに合わせて経路を構築するため，それ以降のブロックが他のノードにより送信されると，その経路で送らないとならないためである．またホップ数が多いほどブロックが届くまでの時間が長くなることから遅延時間が増加する．そのため，ホップ数を抑え，遅延を削減する方法を考える必要がある．

4. まとめ

Plumtree を用いたブロックチェーンネットワークにおけるメッセージの削減の提案及びシミュレーションを評価した．その結果，従来手法よりメッセージ数の削減を見込めることが分かった．今後はホップ数が増えるほど，遅延が発生するためホップ数を削減する仕組みが必要であることが分かった．

今後は，ホップ数を削減するためにスパニングツリーの構築を改良する手法を検討する．

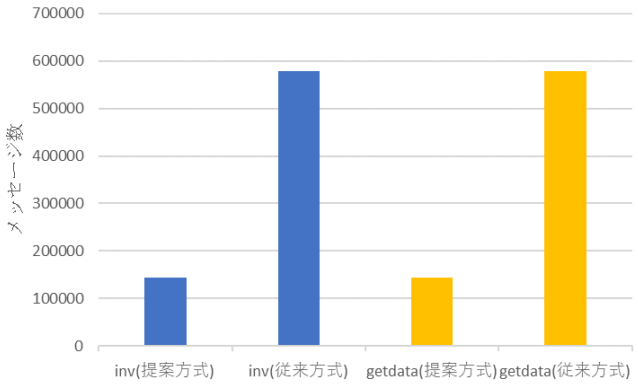

図 2 メッセージ数の比較

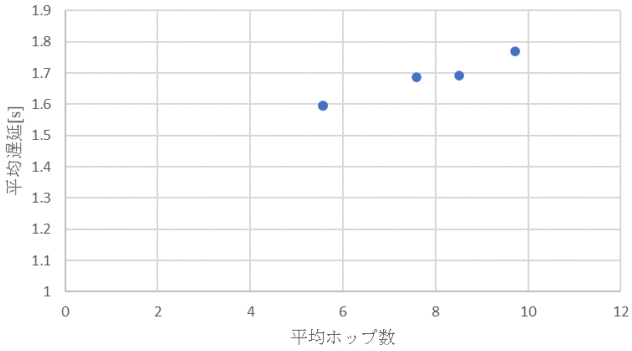

図 3 ホップ数と遅延時間の関係

謝辞　本研究の一部は，日揮・実吉奨学会の支援を受けて行われたものである．

参考文献

[1] Satoshi Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System”, 2008.

[2] João Leitão, Jos Pereira, Luĕıs Rodrigues, “Epidemic Broadcast Trees”, 26th IEEE International Symposium on Reliable Distributed Systems (SRDS 2007), pp.301-310, Oct. 2007.

[3] Andreas M. Antonopoulos 著, 今井崇也, 鳩貝純一郎訳 “Masterring Bitcoin Bitcoin とブロックチェーン 暗号通貨を支える技術”, NTT 出版, 2016.

[4] Yusuke Aoki, Kai Otsuki, Takeshi Kaneko, Ryohei Banno, Kazuyuki Shudo, "SimBlock: A Blockchain Network Simulator", Workshop on Cryptocurrencies and Blockchains for Distributed Systems (CryBlock, in conjunction with IEEE INFOCOM), pp. 325-329, April 2019.

[5] Blockchair, https://blockchair.com/nodes (閲覧日 2020/12/15)

[6] Etherscan, https://etherscan.io/nodetracker (閲覧日 2020/12/15)

[7] BitInfoCharts, https://bitinfocharts.com (閲覧日 2020/12/15)