一般社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
NS2019-186(2020-03)

double sweep に基づく P2P ネットワークの経路長短縮

金子　孟司[†]　坂野　遼平[††]　首藤　一幸[†††]

[†], [††], [†††] 東京工業大学　〒 152–8550 東京都目黒区大岡山 2–12–1
E-mail: †kaneko.t.ay@m.titech.ac.jp, ††banno@computer.org, †††shudo@is.titech.ac.jp

あらまし　P2P ネットワーク上の各クエリに対する経路長を短縮することは，通信時間の短縮や耐故障性の向上という観点で重要な課題となっている．また，double sweep は木の直径を求める古典的なアルゴリズムのひとつで，グラフに対しては直径の下界を与える役割をもつ．本研究では double sweep の動作に基づいたトポロジの改善手法を提案し，複数の参加ノードがそのネットワーク上で broadcast を行う状況下での経路長の短縮を試みる．評価実験によって性質の悪いグラフに対しては大きな改善を，性質の良いグラフに対しては小さい経路長を維持することを確認した．

キーワード　P2P ネットワーク，broadcast，直径，double sweep

Shortening Path Lengths on P2P Networks with a Double Sweep Based Technique

Takeshi KANEKO[†], Ryohei BANNO[††], and Kazuyuki SHUDO[†††]

[†], [††], [†††] Tokyo Institute of Technology　2–12–1 Ookayama, Meguro-ku, Tokyo, 152–8552 Japan
E-mail: †kaneko.t.ay@m.titech.ac.jp, ††banno@computer.org, †††shudo@is.titech.ac.jp

Abstract　Shortening the path lengths for each query on a P2P network is an important challenge in terms of reducing delay time and improving fault tolerance. In this paper, we propose a double sweep based technique to shorten the path lengths by broadcasts on the situation where each node sends broadcasts on the network. The evaluation experiments show that the proposed method performs a large improvement for graphs with poor properties, and maintains short path lengths for graphs with good properties.

Key words　P2P network, broadcast, diameter, double sweep

1. はじめに

大規模な P2P ネットワークの分散システムは，その高いスケーラビリティや信頼性の実現に向けて，多数のノードによる自律分散的な動作の元での高速な情報のやり取りを必要とする．そのため，broadcast を含む様々なクエリに対する経路長や通信時間を短縮することは P2P ネットワークの重要な課題のひとつとなっている．

効率的な broadcast を実現するためのアルゴリズムはこれまで多くの提案がされてきた [1]．本研究は，ネットワークのトポロジ自体を動的に変化させせることで直径を短縮し，broadcast の経路長を短縮することを目指す．

しばしば double sweep と呼ばれる手法 [2] は，ノード数 n に対して $O(n)$ の時間計算量で木の直径を求める古典的なアルゴリズムのひとつである．また，重みなし無向グラフに対しては直径の下界として有効であることが経験的に知られている [3]．そこで私たちは，double sweep の動作に基づいた提案手法によって，複数の参加ノードがそのネットワーク上で broadcast を行う状況下における経路長の短縮を試みる．また，提案手法による追加の通信メッセージが多く発生するのを避けるために，経路上の各ノードが broadcast の処理に便乗するかたちで特定の処理を実行するようにアルゴリズムを設計している．

本稿の構成は以下の通りである．2. で本論文で用いる記法を整理し，3. で提案手法の詳細を述べる．次に 4. で評価実験の内容とその結果を述べ，5. で本研究のまとめと今後の課題について述べる．

2. 記　　法

本論文で用いる記法と定義を以下にまとめる．

有限のノード集合 V とエッジ集合 E で構成される重みなし

This is an unrefereed paper.

無向グラフ G を $G=(V,E)$ と表記する．ただし，多重辺や自己ループはないものとする．このとき，G に対して以下を定義する．

- $\mathrm{adj}_G : V \ni v \mapsto$「$v$ の隣接ノード集合」$\in 2^V$
- $\mathrm{dist}_G : V \times V \ni (u,v) \mapsto$「$u,v$ 間の最短経路長」$\in \mathbb{N} \cup \{\infty\}$
  - 特に ${}^\forall v \in V, \mathrm{dist}_G(v,v)=0$ とする．
- $\mathrm{furthestNodes}_G : V \ni v \mapsto \mathop{\arg\max}_{u \in V}\{\mathrm{dist}_G(v,u)\} \in 2^V$
  - これは v からの経路長が最も大きいノード，つまり v の離心数をなすノードの集合である．これらのノードを「v の最遠ノード」と呼ぶこととする．また，V は有限集合なので ${}^\forall v \in V, \mathrm{furthestNodes}_G(v) \neq \emptyset$ を満たす．
- 発信ノードが $s \in V$ であるネットワーク G 上の flooding による broadcast b を $b=(G,s)$ と表記する．$b=(G,s)$ に対して：
  - $T_b=(V,E_b;s)$：b の実行によって形成される s を根とする無向根付き木．エッジに重みがないため，s を始点とした G 上の BFS（Breadth-First Search）木と等価である．

また，有限のノード集合 V とエッジ集合 E で構成される $s \in V$ を根とする無向根付き木 T を $T=(V,E;s)$ と表記する．単に無向グラフ $T=(V,E)$ とみなすこともある．さらに T に対して以下を定義する．

- $\mathrm{children}_T : V \ni v \mapsto$「$v$ の子ノード集合」$\in 2^V$

3. 提案手法

P2P ネットワークのトポロジ自体を動的に変化させることで直径を短縮し，broadcast の経路長を短縮することを目指す．ただし極端な例として，エッジを自由に追加して完全グラフにするといった自明な方法で直径を短縮することは可能である．しかし，これはスケーラビリティの観点から現実的な手法ではない．そのため以降では，設計するアルゴリズムの制約として「各ノードの次数を一定にする」という条件を設けることにする．

また，提案手法が動作する状況として以下を想定している：

（1） ネットワークは双方向にリンクを張っている，つまり無向グラフとみなせる．

（2） 初期のネットワークがグラフとして連結である．

（3） ネットワークの各ノードが flooding による broadcast を何度も行う．

（4） ネットワークの全ノードが提案手法のプロトコルに従っている．

1，2 番目は議論の単純化のため，3 番目は提案手法が broadcast に便乗するかたちで動作するため，4 番目は現段階では悪意あるノードの存在を考慮して設計していないため，このような想定を設けている．

3.1 トポロジの改善アルゴリズム

大まかな流れは，各ノードが自律分散的に double sweep の処理を模倣することで直径となりうるノードペアを検出してエッジを張るというものになっている．double sweep のアルゴリズムの詳細は付録を参照されたい．

Algorithm 1: The optimization process in $v \in V$ for a network topology $G=(V,E)$

```
initialize :
    $v$.alreadyFurthest ← False
    $v$.target ← None
1  upon receiving broadcast $b=(G,s)$ then
2      $b$ に対する flooding による broadcast の処理を実行する
3      if $\neg s$.alreadyFurthest then
4          if $v \in \mathrm{furthestNodes}_{T_b}(s)$ then
5              $v$.alreadyFurthest ← True
6      else
7          $v$.alreadyFurthest ← False
8          if $v \in \mathrm{furthestNodes}_{T_b}(s)$ then
9              $v$.target ← $s$
10             request $\langle s.\mathrm{target} \leftarrow v \rangle$ to $s$
11     $u \leftarrow v$.target
12     if $u$ = None then
13         return
14     $\hat{v} \leftarrow$ request $\langle u.\mathrm{target} \rangle$ to $u$
15     if $\hat{v} \neq v$ then
16         return
17     $A_v \leftarrow \mathrm{adj}_G(v) - \mathrm{adj}_{T_b}(v)$
18     $A_u \leftarrow$ request $\langle \mathrm{adj}_G(u) - \mathrm{adj}_{T_b}(u) \rangle$ to $u$
19     if $A_v = \emptyset \lor A_u = \emptyset$ then
20         return
21     $v' \leftarrow A_v$ から任意のノードをとる
22     $u' \leftarrow A_u$ から任意のノードをとる
23     request $\langle E \leftarrow E \cup \{\{v,u\}\} \rangle$ to $v, u$
24     request $\langle E \leftarrow E \cup \{\{v',u'\}\} \rangle$ to $v', u'$
25     request $\langle E \leftarrow E - \{\{v,v'\}\} \rangle$ to $v, v'$
26     request $\langle E \leftarrow E - \{\{u,u'\}\} \rangle$ to $u, u'$
27     $v$.target ← None
28     request $\langle u.\mathrm{target} \leftarrow \mathrm{None} \rangle$ to $u$
```

Algorithm 1 は，ネットワーク $G=(V,E)$ 上のノード $v \in V$ が，$s \in V$ を発信ノードとする broadcast $b=(G,s)$ を受信したときの提案手法の処理過程を示した疑似コードである．ここで，「**request** $\langle \mathrm{statement} \rangle$ **to** v」はノード $v \in V$ に対して statement の処理をリクエストすることを意味している．また，「$x \leftarrow$ **request** $\langle \mathrm{expression} \rangle$ **to** v」はノード $v \in V$ に対して expression の計算結果をリクエストして，そのレスポンス結果を x に代入していることを意味している．

受信ノード $v \in V$ は，flooding による通常の broadcast の処理を実行（2 行目）したのち，直径となりうるノードの近似ペアを検出する処理を行う（3–10 行目）．このとき，v.alreadyFurthest の値は「ノード v が過去の broadcast においてすでに最遠ノー

ドとして判定済みであるか」を意味する．つまり，受信した broadcast b は，$\neg s$.alreadyFurthest ならば double sweep における 1 度目の BFS に，s.alreadyFurthest ならば 2 度目の BFS に相当する．2 度目の BFS に相当するとき，発信ノード $s \in V$ と最遠ノード $v \in \mathrm{furthestNodes}_{T_b}(s)$ は double sweep において出力されるノードペアに該当する．つまり，このノードペア間の経路長は G の直径に近い値となることが推測されるため，s, v 間でエッジを張るための予約処理を行う（9–10 行目）．ここで，v.target の値は「v が接続するべきと判定されたノード」を意味する．

続く 11–28 行目は実際にトポロジを修正する処理を行っている．予約されたノードペア (v, u) を検出した場合，エッジの張替え処理を行う．ここで，$A_v, A_u \subset V$ は，ノード v, u それぞれに関して，隣接ノードであるが broadcast b における通信相手としては冗長だったノードの集合を意味する．任意にノード $v' \in A_v, u' \in A_u$ を選んだのち，次のように G のエッジを更新する．

- エッジ $\{v, u\}$ を G に追加する．
- エッジ $\{v', u'\}$ を G に追加する．
- エッジ $\{v, v'\}$ を G から削除する．
- エッジ $\{u, u'\}$ を G から削除する．

エッジの更新の仕方から，各ノードの次数は変化していないことがわかる．また，更新前の G のエッジ集合を E_{prev} としたとき，$\{v, u\}, \{v', u'\} \notin E_{\mathrm{prev}}$ かつ $\{v, v'\}, \{u, u'\} \in E_{\mathrm{prev}} - E_b$ を満たす．つまり，直感的にこのエッジの張替え処理は，直径となりそうなノードペア (v, u) 間のエッジを張り，broadcast 時に冗長となりうるノードペア $(v, v'), (u, u')$ 間のエッジを除去し，次数を変化させないためにノードペア (v', u') 間のエッジを張る，という動作になっている．

本手法が動作する状況としてネットワークの各ノードが broadcast を何度も行うことを想定している．そのため，時間経過とともに徐々に直径が短縮し，合わせて broadcast 時の経路長も短縮することが期待される．

3.2 連結性

トポロジの更新によってネットワークが分断すると，ノード間の連携が現実的に困難になる．そのため，このような事態は回避するべきである．そこで本節では，Algorithm 1 によるトポロジの更新に対する連結性を議論する．

Algorithm 1 においてネットワークから削除するエッジは，flooding による broadcast で冗長であったエッジから選択するように設計されている．そのことから次の命題が成立する．

［定理 1］　ネットワーク $G = (V, E)$ に対して以下の条件を満たすとき，Algorithm 1 によるトポロジの変更はネットワークを分断しない．

- 複数の broadcast の処理がネットワーク全体として同時に行われない．つまり，ある broadcast b に対する処理をあるノードが実行中に，別の broadcast b' が発信されない．
- 任意の broadcast の処理中にノードの参加や離脱が発生しない．

証明　今，G 上の任意の broadcast $b = (G, s)$ が発信されたとする．このとき，実行前後のネットワークのグラフをそれぞれ $G_{\mathrm{prev}} = (V, E_{\mathrm{prev}})$，$G_{\mathrm{next}} = (V, E_{\mathrm{next}})$ とおく．このとき，G_{prev} が連結ならば G_{next} も連結であることを示せば十分である．

G_{prev} が連結であると仮定する．Algorithm 1 の処理中にトポロジの変更が発生しなかったならば $G_{\mathrm{next}} = G_{\mathrm{prev}}$ より G_{next} は連結である．よってトポロジの変更が発生した場合のみを考えればよい．ノード $v, v', u, u' \in V$ が Algorithm 1 内で使用されている変数とそれぞれ同一であるとすると，$E_{\mathrm{next}} = E_{\mathrm{prev}} \cup \{\, \{v, u\}, \{v', u'\} \,\} - \{\, \{v, v'\}, \{u, u'\} \,\}$ が成り立つ．ここで $T_b = (V, E_b)$ は $E_b \subset E_{\mathrm{prev}}$ を満たし，さらに $\{v, v'\}, \{u, u'\} \notin E_b$ であるため，$E_{\mathrm{next}} \supset E_b$ が成り立つ．また，T_b は木なので連結であり，加えて連結グラフ G_{prev} の全域木であるため，$G_{\mathrm{next}} = (V, E_{\mathrm{next}})$ も連結である．□

以上より，定理中の条件を満たすならばネットワークが分断しないことが保証された．

3.3 最遠ノードの判定

ネットワークの帯域やメンテナンスコストを考慮すると，追加のメッセージを最小限に済ませつつ，各ノードが自律分散的に動作するようにするのが好まれる．Algorithm 1 は基本的にそのような設計になっているが，$v \in \mathrm{furthestNodes}_{T_b}(s)$ の判定，つまり自身が発信ノードの最遠ノードになっているかの判定は，現状では各ノードが単独で判定するのは不可能である．

そこで本節では，最遠ノードの判定を少ない追加メッセージによって可能にすることを目指す．例として 2 つの手法を提示する．

3.3.1 broadcast を往復することで情報を得る手法

根付き木 $T = (V, E; s)$ に関して $v \in V$ を根とする部分木を $T^v = (V^v, E^v; v)$ とおくと，${}^\forall v \in V$ に対して

$$\max_{u \in V^v} \{\mathrm{dist}_{T^v}(v, u)\} =$$
$$\max\left(\{\, 0 \,\} \cup \left\{ \max_{u \in V^c} \{\mathrm{dist}_{T^c}(c, u)\} + 1 \;\middle|\; c \in \mathrm{children}_{T^v}(s) \right\} \right)$$

が成り立つ．よって $\max\limits_{v \in V} \{\mathrm{dist}_T(s, v)\}$ と $\arg \max\limits_{v \in V} \{\mathrm{dist}_T(s, v)\}$ $(= \mathrm{furthestNodes}_T(s))$ は再帰的に求まる．

これを利用することで，最遠ノードを判定する単純な手法として次が考えられる．

（1）　broadcast $b = (G, s)$ を実行するときに，s からの経路長 $\mathrm{dist}_{T_b}(s, \cdot)$ の情報を piggyback する．

（2）　broadcast 後に T_b の葉から根に向かって，各ノード $v \in V$ において $\max\limits_{u \in V^v} \{\mathrm{dist}_{T_b^v}(v, u)\}$ と $\mathrm{furthestNodes}_{T_b^v}(v)$ を計算する．

（3）　s が $\mathrm{furthestNodes}_{T_b}(s)$ を計算したのち，各 $v \in \mathrm{furthestNodes}_{T_b}(s)$ に最遠ノードであることを伝える．

この手法の利点は，各ノードが確実に最遠ノードの判定を可能とすることである．一方で欠点は，broadcast を実行するたびに $O(|V|)$ 回の追加メッセージを必要とすることである．

Algorithm 2: The update of v.threshold in $v \in V$ for a network topology $G = (V, E)$

```
initialize :
    $v$.threshold ← 0
upon receiving broadcast $b = (G, s)$ then
    $b$ に対する Algorithm 1 の処理を実行する
    $v$.threshold ← $\max\{0, v.\mathrm{threshold} - 1\}$
    $v$.threshold ← $\max\{0, \mathrm{dist}_{T_b}(s, v)\}$
```

ただし，葉から根に情報を伝播する過程では T_b を辿れば良いため冗長なメッセージは発生しない．そのため，単純に追加の送信メッセージが 2 倍になっているわけではなく，それよりも格段に少ないメッセージ数であることは留意したい．

3.3.2 逐次更新する閾値を使用して判定する手法

次に閾値を使用して判定する手法を提案する．本手法の流れは次のとおりである．各ノード $v \in V$ は閾値を表す $v.\mathrm{threshold} \in \mathbb{N}$ の値をもつものとする．ここで，$s \in V$ を発信ノードとする broadcast $b = (G, s)$ が実行されたときに

$$U_b \coloneqq \{ v \in V \mid \mathrm{children}_{T_b}(v) = \emptyset \land \mathrm{dist}_{T_b}(s, v) > v.\mathrm{threshold} \}$$

を最遠ノードの候補の集合とする．このとき，各ノード $v \in U_b$ は自身が最遠ノードであるかの問い合わせを発信ノード s に対して行うことによって，判定を行う．

この手法は broadcast を実行するたびに $O(|U_b|)$ 回の追加メッセージを必要とする．その上これらはすべて，発信ノード s が関与するメッセージである．また，$\mathrm{furthestNodes}_{T_b}(s) \cap U_b \neq \emptyset$ ならば正しい最遠ノードの判定が可能である．つまり，$\mathrm{furthestNodes}_{T_b}(s) \cap U_b \neq \emptyset$ を満たしつつ $|U_b|$ を小さくするような v.threshold の更新方法が期待される．

例えば Algorithm 2 のような更新方法が考えられる．直感的にこれは，$\max_{v \in V}\{\mathrm{dist}_{T_b}(s, v)\}$ が，過去の broadcast における発信ノードからの経路長の最大値よりも大きいという推測から設計されている．また，broadcast するたびに v.threshold をデクリメントしている理由はトポロジの改善とともに適切な v.threshold の値が小さくなることから行っている．

4. 評価

提案手法をシミュレーションすることで評価実験を行い，直径や broadcast に対する経路長の変化を観察した．シミュレーションの動作手順は次のとおりである．

（1） 対象となるネットワークのグラフ $G = (V, E)$ を生成する．
（2） この時点での G の各指標を測定する．
（3） 次を 10000 回繰り返す．
  （a） 発信ノード $s \in V$ を一様乱数で決定する．
  （b） Algorithm 1 に従い，s を発信ノードとする broadcast を実行する．
  （c） この時点での G の各指標を測定する．

ただし，ここでは各ノードが最遠ノードの判定を可能であるものとしてシミュレーションを実行した．

また，評価対象とする初期のグラフとして次の 4 つを用いた．各グラフは条件を揃えるために，ノード数 $|V| \simeq 1000$，エッジ数 $|E| \simeq 4000$ になるように設定している．

- $m \times n$ グリッドグラフ：これは，$m \times n$ 個のノードが格子状に並び，各ノードが上下左右のノードとエッジを張ったグラフである．形式的には $m, n \in \mathbb{N}$ に対してノード集合 $V' = \mathbb{Z}/m\mathbb{Z} \times \mathbb{Z}/n\mathbb{Z}$，エッジ集合 $E' = \{ \{(i,j),(i,j{+}1)\} \mid i \in \mathbb{Z}/m\mathbb{Z} \land j \in \mathbb{Z}/n\mathbb{Z} - \{n{-}1\} \} \cup \{ \{(i,j),(i{+}1,j)\} \mid i \in \mathbb{Z}/m\mathbb{Z} - \{m{-}1\} \land j \in \mathbb{Z}/n\mathbb{Z} \}$ のグラフ $G' = (V', E')$ に同型なグラフである．本実験では 32×32 グリッドグラフを用いた．以下，このグラフを `Grid Graph` と表記する．
- $m \times n$ トーラスグリッドグラフ：これは，$m \times n$ グリッドグラフに対し，上端と下端のノードどうし，左端と右端のノードどうしでエッジを張ったグラフである．形式的には $m, n \in \mathbb{N}$ に対してノード集合 $V' = \mathbb{Z}/m\mathbb{Z} \times \mathbb{Z}/n\mathbb{Z}$，エッジ集合 $E' = \{ \{(i,j),(i,j{+}1)\} \mid i \in \mathbb{Z}/m\mathbb{Z} \land j \in \mathbb{Z}/n\mathbb{Z} \} \cup \{ \{(i,j),(i{+}1,j)\} \mid i \in \mathbb{Z}/m\mathbb{Z} \land j \in \mathbb{Z}/n\mathbb{Z} \}$ のグラフ $G' = (V', E')$ に同型なグラフである．本実験では 32×32 トーラスグリッドグラフを用いた．以下，このグラフを `Torus Grid Graph` と表記する．
- Erdös-Rényi モデル [4] によるランダムグラフ：グラフ $\Gamma_{n,N}$ は，ノード数 n を固定したのち $\binom{n}{2}$ から N 個のエッジをランダムに追加して生成されるグラフである．本実験では，ノード数 $|V| = 1000$，エッジ数 $|E| = 4000$ である $\Gamma_{1000,4000}$ のモデルに従って生成した．以下，このグラフを `Random Graph` と表記する．
- Barabási-Albert モデル [5] によるランダムグラフ：これは scale-free 性を満たすグラフの生成モデルのひとつである．本実験では，ノード数を 1000，生成の毎ステップで追加するエッジ数を 4 としてこのモデルに従ってグラフを生成した．以下，このグラフを `BA Model Graph` と表記する．

実験において前者の `Grid Graph` と `Torus Grid Graph` は性質の悪いグラフとして用いている．`Grid Graph` と `Torus Grid Graph` の直径はそれぞれ $62, 32$ でありノード数 1024 に対して比較的大きく，直径の改善の余地が十分にあると考えられる．一方で後者の `Random Graph` と `BA Model Graph` は前者のグラフと対比するための性質の良いグラフとして用いている．ここで性質の良し悪しとは，直径の大小や平均経路長の大小を指している．

図 1，2，3 は実験の結果である．横軸はすべて broadcast の実行回数を表している．図 1 の縦軸は，各ステップにおけるその時点でのグラフの直径を表している．図 2 の縦軸は，各ステップにおけるその時点で，全ノードそれぞれを発信ノードとする broadcast に対する平均経路長の平均値を表している．形式的には

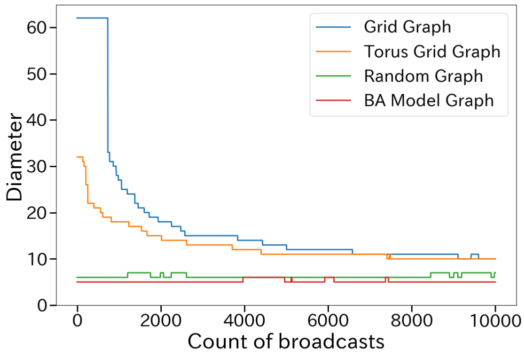

図 1　グラフの直径の変化．
Fig. 1　The diameters of graphs.

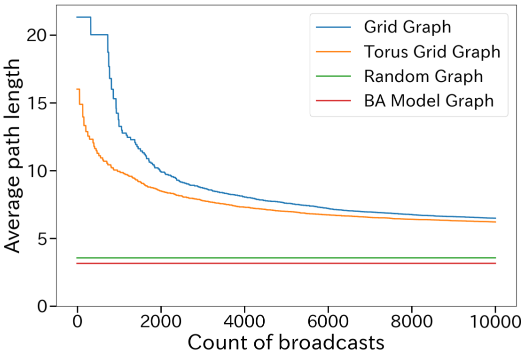

図 2　broadcast による平均経路長の変化．
Fig. 2　Mean of the average path lengths by broadcasts.

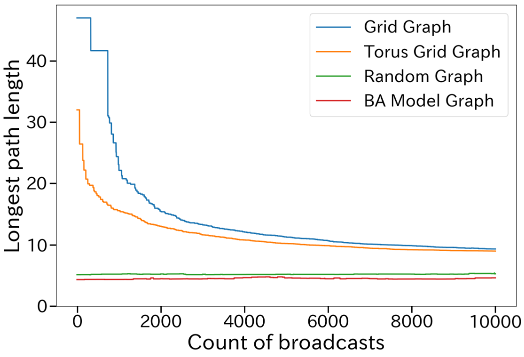

図 3　broadcast による最大経路長の変化．
Fig. 3　Mean of the longest path lengths by broadcasts.

$$\frac{1}{|V|}\sum_{s\in V}\left(\frac{1}{|V|}\sum_{v\in V}\mathrm{dist}_{T_{(G,s)}}(s,v)\right)$$

である．図 3 の縦軸は，各ステップにおけるその時点で，全ノードそれぞれを発信ノードとする broadcast に対する最大経路長の平均値を表している．形式的には

$$\frac{1}{|V|}\sum_{s\in V}\max_{v\in V}\left\{\,\mathrm{dist}_{T_{(G,s)}}(s,v)\,\right\}$$

である．

`Grid Graph` と `Torus Grid Graph` は broadcast の回数が増えるに連れて各指標のいずれの値も小さくなっている傾向が見える．また，その減少量は徐々に緩やかになっている．具体的には，`Grid Graph` では直径，平均経路長，最大経路長がそれぞれ 10，6.5，9.3 付近で，`Torus Grid Graph` では 10，6.2，9.0 付近で落ち着いている．一方で `Random Graph` と `BA Model Graph` に関しては，broadcast の回数に関係なく各指標の変化は乏しい．

以上の結果から提案手法は，性質の悪いグラフに対しては経路長の短縮に大きく貢献し，すでに性質の良いグラフに対しては経路長に与える影響は少なくトポロジが変化しにくいと考察した．

5.　まとめと今後の課題

本研究では double sweep の動作に基づいた提案手法によって，P2P ネットワーク上の複数の参加ノードが broadcast を行う状況下での経路長の短縮を試みた．多数のノードが自律分散的に動作するようにアルゴリズムを設計することで既存の P2P ネットワークに対して適用できるように工夫した．また，提案手法によるネットワークの連結性についての議論も行い，特定の条件下ではネットワークが分断しないことを数学的に証明した．

さらに評価実験によって，性質の悪いグラフに対しては経路長を大きく短縮し，徐々にその減少量は緩やかになりつつ，ある経路長付近で落ち着くことを確認した．また，すでに性質の良いグラフに対しては経路長が小さい状態で維持していることを確認した．実際に提案手法を既存のネットワークに適用する場合，そのネットワークのトポロジ全体を把握するのは困難で未知な部分が多いことがしばしばある．しかし，提案手法による経路長の悪化の見込みはなく改善する可能性があることを考慮すると，経路長短縮のための有効な選択肢のひとつだと考えられる．

一方で，まだ考慮すべき問題は多く存在している．例えば，提案手法を適用することによるメッセージ数の増加やメンテナンスコストの増加，さらにはノードの参加や離脱のある動的なネットワークに対する耐故障性等の考察や対処である．最遠ノードの判定に対する手法も議論の余地は多分にある．その他，既存の broadcast アルゴリズムへの適用や，提案手法の適用によるネットワークトポロジの収束性に関する考察も検討している．

また，現実の大規模な P2P ネットワークの多くは small-world 性があり経路長の改善の余地は小さいという懸念もある．実際，既存の P2P ネットワークに対する small-world 性の例はいくつか唱えられている [6], [7]．4. の評価実験では性質が極端に悪いグラフと極端に良いグラフを用いていたため，より現実の P2P ネットワークに即したグラフに対する提案手法の影響は評価するべきと考える．

謝辞　本研究の一部は，国立研究開発法人新エネルギー・産業技術総合開発機構（NEDO）の委託業務として行われました．本研究は（公財）セコム科学技術振興財団 一般研究助成の支援を受けたものです．

文　献

[1] P. Ruiz and P. Bouvry, "Survey on Broadcast Algorithms for Mobile Ad Hoc Networks," ACM Computing Surveys (CSUR), vol.48, no.1, pp.8:1–8:35, July 2015.

[2] G.Y. Handler, "Minimax Location of a Facility in an Undirected Tree Graph," Transportation Science, vol.7, no.3, pp.287–293, Aug. 1973.

[3] C. Magnien, M. Latapy, and M. Habib, "Fast Computation of Empirically Tight Bounds for the Diameter of Massive Graphs," J. Exp. Algorithmics, vol.13, pp.10:1.10–10:1.9, Feb. 2009.

[4] P. Erdös and A. Rényi, "On random graphs I.," Publicationes mathematicae, vol.6, no.26, pp.290–297, 1959.

[5] A.-L. Barabási and R. Albert, "Emergence of Scaling in Random Networks," Science, vol.286, no.5439, pp.509–512, Oct. 1999.

[6] D. Stutzbach, R. Rejaie, and S. Sen, "Characterizing Unstructured Overlay Topologies in Modern P2P File-Sharing Systems," IEEE/ACM Transactions on Networking, vol.16, no.2, pp.267–280, April 2008.

[7] M.A. Javarone and C.S. Wright, "From Bitcoin to Bitcoin Cash: A network analysis," Proceedings of the 1st Workshop on Cryptocurrencies and Blockchains for Distributed Systems, pp.77–81, CryBlock'18, Association for Computing Machinery, Munich, Germany, June 2018.

付　録

1. Double Sweep

Algorithm 3 は重みなし無向グラフ $G = (V, E)$ に対する double sweep と呼ばれるアルゴリズムの疑似コードである．$\mathrm{furthestNodes}_G(u)$ と $\mathrm{furthestNodes}_G(v)$ を求める部分で BFS をすることで時間計算量，空間計算量ともに $O(|V|)$ で動作する．G が木ならば $\mathrm{dist}_G(v, w)$ は G の直径に等しい．

Algorithm 3: Double Sweep

```
Input   : 空でない重みなし無向グラフ $G = (V, E)$
Output  : ノードペア
$u \leftarrow V$ から任意のノードをとる
$v \leftarrow \mathrm{furthestNodes}_G(u)$ から任意のノードをとる
$w \leftarrow \mathrm{furthestNodes}_G(v)$ から任意のノードをとる
return $(v, w)$
```