マルチメディア通信と分散処理 112－2
（２００３．３．２０）

オーバレイマルチキャストにおける遅延最小木の自律的再構築アルゴリズムの提案と評価

常村 和史†, 山口 弘純†, 東野 輝夫†
† 大阪大学 大学院情報科学研究科

本稿では，参加者が動的に変化しやすく，かつ，インタラクティブなグループ通信を行うアプリケーションを対象に，オーバレイネットワーク上で自律的にマルチキャスト木を再構築するアルゴリズムを提案する．参加者が動的に変化する場合，マルチキャスト木の最大遅延時間 (*diameter*) をなるべく小さくすること，および，ホストの離脱に対して，残されたマルチキャスト木の形状をできるだけ維持しながら短い時間でマルチキャスト木を再構築できることが望まれる．提案手法では，２ホップまでで到達可能なノードの接続関係と最大遅延時間情報を各ノードが自律的に管理することにより，各ホストの離脱時に高速に最大遅延時間の小さなマルチキャスト木が再構築できるように工夫している．シミュレーション実験により，ホストの参加・離脱のたびに静的にマルチキャスト木を構築する従来手法と比べ，提案アルゴリズムでは，最大遅延時間をほとんど増大させることなく，木の形状をなるべく維持したマルチキャスト木を短時間 (従来手法の 1/4〜1/20 の時間) で再構築できることが分かった．

オーバレイネットワーク，アプリケーションレベルマルチキャスト，分散アルゴリズム，遅延最小木

An Autonomous Algorithm for Reconstructing Minimum Maximum-Latency Trees in Overlay Networks

Kazufumi Tsunemura†, Hirozumi Yamaguchi† and Teruo Higashino†
† Graduate School of Information Science and Technology, Osaka University

In this paper, we propose an autonomous reconstruction algorithm of minimum maximum-latency trees in overlay networks. The algorithm can be used for applications whose participants tend to change dynamically and communicate with each other interactively. In those applications, it is desired that the diameters of virtual overlay multicast trees are made as small as possible, and that for any participant's leaving, a new multicast tree is reconstructed for a short time while maintaining its tree structure. In the proposed algorithm, each node holds the information about the neighboring nodes reachable with at most two hops. When one intermediate node is left, its neighboring nodes exchange their disconnected sub-trees' diameter information and reconstruct a new multicast tree autonomously. From the simulation results, it is shown that the proposal algorithm can reconstruct similar diameter's multicast trees with very small costs, and that the obtained new multicast trees maintain the previous multicast trees' structure as much as possible compared with the existing centralized algorithm.

Overlay Network, Application Level Multicast, Decentralized Algorithm, Minimum Maximum-Latency Tree

1 はじめに

ビデオ会議システムやファイル配信など，多数のホストにデータ配信を行うアプリケーションではマルチキャスト通信が有用である．IP マルチキャストは，ネットワーク資源の利用効率という点では非常に有用であるが，現在のインターネットでのインフラストラクチャの普及の観点から広域通信の実現は容易でないという問題点が挙げられる．一方，アプリケーションレベルマルチキャストは，ホスト間ユニキャストを仮想リンクとみなしたオーバレイネットワーク上でのマルチキャスト (オーバレイマルチキャスト) である．これは，IP マルチキャストと比較して，現在のインフラストラクチャでの実現が容易であり，リンク制御機能として既存のユニキャストプロトコルが利用可能であるなどの利点がある．その一方で，ホスト付近での帯域制約や複数のユニキャストコネクションを経由することによる遅延時間の増大，さらに，ホストの離脱時におけるネットワーク再構築のオーバヘッドを考慮し，適切な制御を行う必要がある．

アプリケーションレベルマルチキャストに関する従来研究では，ホストの離脱などを考慮した動的なマルチキャスト木構築を対象としたものや，静的な木構築のみを対象としたものが多数提案されている [2, 3, 4, 5, 6, 7, 8, 9]．しかしその多くは，そのマルチキャスト木構築を集中的なアルゴリズムによって実現しているためにスケーラビリティの面で問題がある．また，単一ソースノードからのマルチキャスト木構築に限定しているものが多い．例えば，文献 [8] ではホストの参加・離脱が動的に発生する環境における小規模グループ向けのマルチキャスト木の構築を集中的に行うアルゴリズムを提案している．この文献では，各リンクの遅延時間の総和を最小にする木の構築を行っている．また，文献 [7] では，ストリーミングのような遅延時間の制約に厳しいリアルタイムアプリケーションを対象にしたアルゴリズムを提案しているが，この文献では単一ソースノードからの平均遅延時間が最小となる木の構築を動的に行っている．

本稿では，マルチキャストアプリケーションとして，ビデオチャットシステムなど性能が遅延に影響されやすく，参加者が動的に変化しやすいインタラクティブなグループ通信アプリケーションを対象としたオーバレイマルチキャスト木構築アルゴリズムを提案する．そのようなオーバレイマルチキャスト木は，ホストの参加・離脱に伴う木の再構築が頻繁に要求されることから，管理が容易な共有木 (被覆木) であることが望ましい．また，インタラクティブなアプリケーションで利用されるため，特定のホストからの遅延時間ではなく，マルチキャスト木上の最大遅延時間 (以下 *diameter* と呼ぶ) がなるべく小さくなることが望ましい．したがって，提案手法ではマルチキャスト木最大遅延時間がなるべく小さい共有木をホストの離脱ごとに少ない計算量で再構築することを目的とする．その際ホスト付近の帯域制約を考慮し，各ホストに次数制約を与え，その制約を満たす共有木を構築するようにする．この次数制約付き共有木の *diameter* 最小化問題は，NP 完全であることが知られている

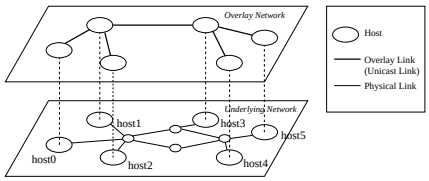

図 1: オーバレイネットワークの構成

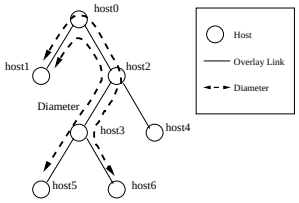

図 2: 木の $diameter$

[1]．従来研究の中で，この問題を対象としているものとして文献 [2] が挙げられる．しかし，文献 [2] の手法はホストの参加，離脱を考慮せず，静的に共有木を構築する集中制御型のアルゴリズムであるため，ホストの離脱ごとに新たに木を構築しなおす必要があり，本研究で対象としているアプリケーションには適当ではない．さらに，提案アルゴリズムでは木の動的再構築をデータの集中管理なしに行う．このために，各ホストは隣接ホストと，さらにその隣接ホスト (高々２ホップで到達可能なホスト) のアドレスのみを保持する状態で，木の分断を検知した場合にホスト間で情報を交換しながら木の再構築を行う．

提案アルゴリズムの評価のために，本研究と同様のメトリックを採用している文献 [2] のアルゴリズムを比較対象としたシミュレーションを行った．なお，このアルゴリズムの時間計算量は，セッションに参加しているホスト数を n とすると，$O(n^3)$ である．

本稿の構成は以下の通りである．２章では本稿で対象としている問題について述べる．３章では，同問題に対する提案アルゴリズムについて述べる．４章では，提案アルゴリズムのシミュレーション実験による評価について述べる．最後に５章でまとめと今後の課題を述べる．

2　次数制約付き $diameter$ 最小化問題

本章では，本稿で対象としているネットワーク，および，それに対する問題について述べる．ここで，図１に本研究で対象とするオーバレイネットワークの構成を示す．このオーバレイネットワークに対して，本研究で対象とする次数制約付き共有木の $diameter$ 最小化問題の概要は以下の通りである．

入力として無向完全グラフ $G = (V, E)$，セッションに参加しているホストの集合 $V = \{v_i | 1 \leq i \leq |V|\}$，各ホスト v_i ごとの次数制約 $d_{max}(v_i)$，および，ホスト i，j 間を結ぶ辺のコスト $c(i, j)$ を与えたときに，構築するマルチキャスト木 T の $diameter$ を $dia(T)$，T に含まれる各ホスト v_i の次数を $d_T(v_i)$ とすると，$d_T(v_i) \leq d_{max}(v_i)$ を満たし，かつ，$dia(T)$ が最小となる T を求める問題である．ここでマルチキャスト木の $diameter$ とは，その木に含まれるホスト間の経路の中で最長となる経路の長さを指し，その経路自体を以下では $diameter$ 経路と呼ぶ．例えば，図 2 においては，$< host1, host0, host2, host3, host5 >$，または，$< host1, host0, host2, host3, host6 >$ がその木の $diameter$ 経路となり，各辺のコストを１と仮定すると $diameter$ は４である．

ビデオ会議システムやファイル配信等のインタラクティブなアプリケーションでは，特定のホストからの遅延時間ではなく，各ホスト間の最大遅延時間がなるべく小さくなることが望ましい．したがって，本稿では $diameter$ がなるべく小さい共有木の構築を行う．また，そのようなアプリケーションでは，ホストはマルチキャストセッションに動的に参加・離脱するものと考えられる．提案手法では，ある単位時間間隔 (これをタイムスロットと呼ぶ) ごとにホストの参加，あるいは，離脱が発生するものと仮定する．これによりネットワークの構成が変化するため，マルチキャスト木の再構築が必要となってくる．ここで，時刻 t におけるホストの集合を $D_t = \{v_i | 1 \leq i \leq |D_t|\}$，構築されるマルチキャスト木を T_t とすると，本研究で対象とする問題は次のように定義される．

入力として無向完全グラフ $G = (V, E)$，各頂点 (ホスト) $v_i \in V$ に対して，次数制約 $d_{max}(v_i)$，および，頂点 $v_i, v_j \in V$ 間の辺 $e_{ij} \in E$ に対して，辺のコスト $c(e_{ij}) > 0$ を与える．

その元で出力として，時刻 t において，$\forall v_i (1 \leq i \leq |D_t|)$ について，$d_{T_t}(v_i) \leq d_{max}(v_i)$，かつ，$dia(T_t) \rightarrow min$ となる T_t を得る．

3　提案アルゴリズム

本章では，オーバレイネットワークにおけるマルチキャスト木動的再構築問題に対する提案アルゴリズムについて述べる．提案アルゴリズムでは集中サーバを仮定することなく木の構築を行う．ここで，各ホストで利用可能な情報は，そのホストと隣接するホスト，および，その隣接ホストの情報 (高々２ホップ分の情報) のみと仮定する．よって，それ以外の情報 (例えば，木全体の $diameter$ 等) を各ホストが得ようとする場合は，メッセージ交換により情報を得る必要がある．分散アルゴリズムではこの情報を集めるのに必要とされるメッセージ量をできるだけ小さくすることが重要である．したがって提案アルゴリズムでは，そのメッセージ量をできるだけ抑えながら効率的なマルチキャスト木構築を行う．さらに，マルチキャスト木の再構築を行う際に必要となる，各ホスト間の再接続はできるだけ少ない方が望ましいと考えられる．そのため，できるだけ現在構築されている木の形状を維持したまま再構築を行う．なお，提案アルゴリズムでは，木構造の管理を容易にするために隣接ホスト間に親子関係を与える．これは MAODV[12] などでも仮定しており，データの配信方向を規定するものではない．また，既にマルチキャスト木に参加しているホストのリストは，あるサーバ (ロビーサーバ) が管理していると仮定する．

アルゴリズムでは，マルチキャストセッションを

1. セッション開始前
2. セッション中のホスト参加時
3. セッション中のホスト離脱時

の３つの段階に分割して，それぞれの構築，及び，再構築における手順について説明する．

3.1　セッション開始前

セッション開始前にマルチキャスト木を構築する方法としては，以下の 2 つの方法が考えられる．

1. 参加要求をしたホストから順に木を構築．
2. ある一定数まで要求を受け付けて，集中的なアルゴリズムで初期木を構築．

これら２つの方法を比べた場合，初期に構築される木の $diameter$ は２．の方法で構築する方がよい結果が得られると考えられる．しかし，本研究では，各ホストが自律的に木を構築し，できるだけ集中的な処理を避けることを目標としているため，１．の方法を用いて初期木を構築することにする．シミュレーションでは，１．の方法と，２．の方法として初期木を文献 [2] のアルゴリズム (従来アルゴリズムと呼ぶ) を用いて構築した場合との比較を行い，初期木が提案アルゴリズムに及ぼす影響についての考察を行っている．

セッション開始前には，各ホストはロビーサーバから既にマルチキャスト木に参加しているホストのリストを受信する．次に，それらのホストとの遅延を計測し，最も遅延時間の小さいホストから順に接続要求を出す．接続要求を受け付けたホストは，自身の次数制約を満たす場合はその要求を受け入れる．この操作を接続要求が受け入れられるまで行う．

3.2 ホスト参加時

セッションの途中であるホストが参加要求を出した場合は，セッション開始時と同様に，サーバからすでにマルチキャスト木に参加しているホストのリストを受信し遅延時間の小さいホストから順に接続要求を出す．この方法で新たにホストを木に追加する場合，遅延時間の小さいホストに順に接続をしていくことになるため，木全体のトポロジーを利用して適切な接続ホストを計算する場合に比べ，$diameter$ の増加が大きくなってしまう可能性が考えられる．しかし，実際には最も遅延時間の小さいホストと接続を行うので $diameter$ の増加は比較的抑えられる (詳細は 4 章参照)．

3.3 ホスト離脱時

マルチキャスト木上のホストが離脱した場合，そのマルチキャスト木が複数の部分木に分割されることになる．このとき，各部分木の中心付近のホスト間で接続を行うことによって $diameter$ を小さくすることが可能である．ここで木の中心のホストとは，$diameter$ を実現する経路上に存在し，かつ，葉ホストからの距離が $diameter$ の $1/2$ になるようなホストを意味する．よって，その部分木の中心付近のホストを選択するために各部分木の $diameter$ を知る必要がある．また，各部分木の $diameter$ や次数の情報をどこでどのように管理するかによって再構築に要する時間やメッセージ量に大きく関わってくるものと考えられる．提案アルゴリズムでは以下の手順で木の再構築を行う．

1. 部分木を管理するホストへの情報集約
2. 部分木の $diameter$ の計算
3. 解候補の選択
4. 接続するリンクの決定

それぞれについてのプロトコルを与える．

3.3.1 部分木の管理

あるホストの離脱によってマルチキャスト木が部分木に分割された場合，できるだけ早く木の再構築処理を行う必要がある．そのため，その処理が素早く行えるように，各部分木の管理は，離脱したホストに接続していたホストが行うものとする．このとき，各部分木の $diameter$ 情報や再接続の際に候補となるホストの情報をこの管理ホストに集めるために，このホストを部分木の根とするのがメッセージ量を削減するために有効であると考えられる．ここで，離脱したホストの子ホストに関しては元々その部分木の根となっているが，親ホストに関しては部分木の根となっていないために親子関係を入れ換える必要がある．よって親ホスト側の部分木に対して，親子関係の入れ換えをホストが離脱した直後に行う．また，部分木間を再接続する場合には，各部分木の管理ホストが集めた情報をあるホストに集約し，再構築計算を行うとする．具体的には，離脱したホストの親ホストがこれを行うとする．例えば，図 4 において，離脱したホスト $host4$ の子ホスト $host7$，$host8$ は，$host4$ の親ホスト $host1$ に各部分木の情報を集約し，木の再構築計算を行う．

3.3.2 部分木の $diameter$ の計算

各ホスト $i(1 \leq i \leq |D_{T_s}|)$ で保持している $diameter$ の情報は次の通りである．ここで，ある部分木 T_s に対し，その部分木に含まれるホスト集合を D_{T_s} で表すとする．

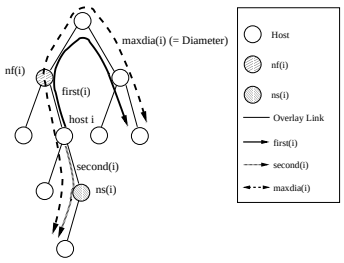

図 3: $diameter$ の情報

- ホスト i からの各部分木 T_s 上の最大経路長のうち，最長の経路長 $first(i)$ と 2 番目の経路長 $second(i)$．
- 最大経路長 $first(i)$ を実現する経路上にあり，かつ，ホスト i に隣接するホスト $nf(i)$，および，$second(i)$ を実現する経路上にあり，かつ，ホスト i に隣接するホスト $ns(i)$．
- ホスト i を根とする木の最大の $diameter$，$maxdia(i)$．

また，これらの関係を図 3 に示す．

そこで各部分木の $diameter$ の計算は以下のようにして行う．ここでホスト i の親ホストを $p(i)$，ホスト i の子ホスト集合を $C(i)$，ホスト i，j 間のコストを $cost(i,j)$ と表すことにする．

1. 部分木のすべてのホスト i に対して，以下のように初期化を行う．
   - $first(i) = 0$，$nf(i) = i$
   - $second(i) = 0$，$ns(i) = i$
   - $maxdia(i) = first(i) + second(i)$
2. ホスト i を部分木の各葉ホストとする．
3. ホスト i から親ホスト $p(i)$ に，以下の情報を送信する．
   - $first(i) + cost(i, p(i))$
   - $maxdia(i)$
4. $k = p(i)$ とおく．ホスト k はすべての $j \in C(k)$ から情報を受け取り，$j \in C(k)$ の中で $first(j) + cost(j,k)$ が最も大きくなるホスト j_1 と，2 番目に大きくなるホスト j_2 を計算する．また k の $diameter$ の情報を以下のように更新する．
   - $first(k) = first(j_1) + cost(j_1, k)$，$nf(k) = j_1$
   - $second(k) = first(j_2) + cost(j_2, k)$，$ns(k) = j_2$
   - もし，$maxdia(k) > first(k) + second(k)$ なら，$maxdia(k) = first(k) + second(k)$
5. $i = k$ として 3．4．の操作を部分木の根ホストに到達するまで繰り返す．

上記の操作を終えると部分木の根ホスト r の $maxdia(r)$ がその部分木の $diameter$ となる．そこでこの情報をその部分木のすべてのホストに通知するために，根ホスト r はすべての子ホストに対してこの情報を送信し，各ホストが持つ $diameter$ を更新する．これは以下のようにして行う．

1. 根ホスト r の子ホスト $i \in C(r)$ に対して，$maxdia(i) = maxdia(r)$
   - もし，$i = nf(r)$ であれば，$first(i)$，$second(i)$ をそれぞれ $\{first(i), second(i), second(r) + cost(i,r)\}$ の中で最も大きいもの，2 番目に大きいものに更新．同様に，$nf(i)$，$ns(i)$ も更新．

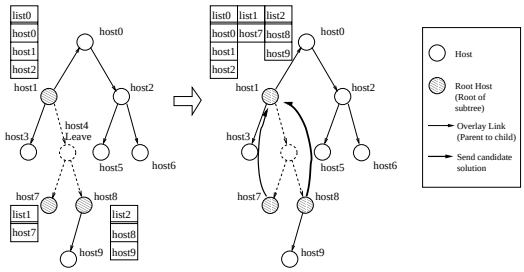

図 4: 解候補の集約

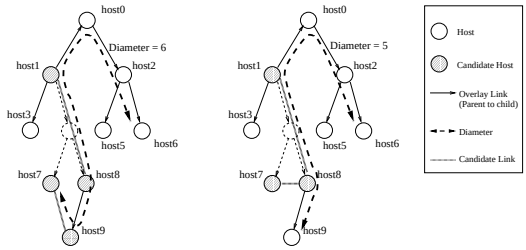

図 5: 解候補の限定

- それ以外であれば，$first(i)$，$second(i)$ をそれぞれ $\{first(i), second(i), first(r)+cost(i,r)\}$ の中で最も大きいもの，２番目に大きいものに更新．同様に，$nf(i)$，$ns(i)$ も更新．

2. $i = r$ として，上記の操作を葉ホストに到達するまで繰り返す．

3.3.3 解候補の選択

上記のように部分木の $diameter$ の情報を受信することによって，各ホストは自分が部分木のどのあたりに位置しているのかがわかる．具体的には，$\frac{|first(i)-dia(T)/2|}{dia(T)/2}$ を計算することにより部分木の中心からどの程度の位置にいるかがわかる．この値が０に近ければそのホストが部分木の中心に近いということを意味し，１に近ければ部分木の葉に近いということを意味する．よって，この値がある定数 $\alpha(0 \leq \alpha \leq 1)$ より小さいかどうかを計算し，小さければその部分木の管理ホストに通知する．また，上記の条件のみでは次数制約が満たされない場合がある．部分木を接続する際は次数制約を満たす必要がある．よって，$\beta_1 \leq d_{max}(i) - d_T(i) \leq \beta_2,\ \ (\beta_1 \leq \beta_2 \leq \max_{i \in T}(d_{max}(i)))$ を満たすホストも候補として部分木の管理ホストに通知する．以上のようにして，各部分木の管理ホストに解候補の情報が集められる．しかし，以上の条件のみでは場合によっては部分木内の多数のホストが解候補となり，解の探索を行う際に非常に時間がかかってしまう可能性が考えられる．そこで管理ホストは，通知してきたホストから定数個のみを選択することで解候補の限定を行う．

3.3.4 解の探索

各部分木で条件を満たすホストが各部分木の根ホストに集められると，そこから最適な経路を計算するためにその情報を１つのホストに集約して計算する必要がある．そこで，3.3.1 で述べたように離脱したホストの親ホストにこれらの情報を集める (図 4)．

以上のようにして，１つのホストに解候補の情報を集めることができる．

次に，どのように解の探索を行うかについて述べる．部分木間の接続は各部分木ごとに１つのホストを選択し，部分木の再構築を行う (図 5 右図)．これは図 5 に示すように，１つの部分木から複数のホストを選択して再構築を行う場合 (図 5 左図) より，得られるマルチキャスト木の $diameter$ を小さくすることが可能であるためである．

以上のようにして，選択された解候補の中から効率良く解空間を限定することが可能である．

次に，限定された解候補の中から最適解の計算を行う．本稿では，最適解の計算を組合せ全探索的に行うものとする．ここで，最適解を求める際の各候補間の遅延時間については，集約された解候補のリストからそれらのホストに問い合わせることにより得られるものとする．

3.4 提案アルゴリズムの時間計算量

ここで，ホスト参加時，ホスト離脱時のそれぞれの場合について，アルゴリズム全体の時間計算量，マルチキャスト木構築に要するメッセージ量，および，メッセージ遅延の評価を行う．ここで，セッションに参加する全ホスト数を n，各ホストの次数制約の最大値を d_{max} とする．

3.4.1 ホスト参加時

以下にホストが新たにセッションに参加した場合の提案アルゴリズムの時間計算量，および，ホスト間で通信されるメッセージ量の評価を与える．ここで，セッションに参加済のホスト数を m $(1 \leq m < n)$ とおく．時間計算量について，参加するホストは参加済のすべてのホストと遅延の計測を行うのでこれを逐次的に行った場合には $O(m)$ となる．さらに，その中で次数制約を満たし，かつ，遅延の最も小さいホストに接続するために $O(m)$ の計算量が必要となる．よって，合計の計算量は $O(m)$ となる．

１ホストあたりのメッセージ量は，遅延を計測する際にサイズが定数オーダーのメッセージを m 個のホストに対して送信するため $O(m)$ となる．また，遅延の最も小さいホストに接続する際も，サイズが定数オーダーのメッセージを最悪 m 個のホストと交換することになるため $O(m)$ となる．よって，合計のメッセージ量は $O(m)$ となる．

3.4.2 ホスト離脱時

ホストがセッションから離脱した場合の提案アルゴリズムの時間計算量，ホスト間で通信されるリンクあたりのメッセージ量，および，メッセージ遅延の評価を行う．

各ホストの提案アルゴリズムでの時間計算量は，そのホストが部分木の根ホストかどうかで変化する．根ホストの場合，部分木全体から集められた定数個の解候補の中からさらに定数個選択するため，時間計算量は $O(1)$ である．また，選ばれた解候補の中から最適解を計算するための時間計算量は，部分木間を接続する辺の組合せ総数ということになる．ここで，各部分木から選択する解候補の数を k とおくと，その組合せ総数は $k^{d_{max}} \frac{(d_{max}(d_{max}-1)/2)!}{(d_{max}-1)!((d_{max}-1)(d_{max}-2)/2)!}$ となる．根ホスト以外のホストの場合，各ホストが計算するのは隣接するホストから受信した $diameter$ の情報から，自身の $diameter$ を更新し，それが条件をみたすかどうかのみであり，また，その情報は定数回の受信のみであるから，計算量は $O(1)$ である．

各リンクあたりのメッセージ量については，まず，$diameter$ を計算する際は，定数オーダーのメッセージを送受信し，さらに隣接するホスト数 (高々d_{max}) に比例するのみであるため $O(1)$ である．解候補の選択についても，各ホストは定数オーダーのリストの送受信を行うのみであるため，リンクあたり $O(1)$ である．また，解の探索においては，これは１つのホストで最適解を計算するためにメッセージ量は０である．さらに，マルチキャスト木を再構築する場合も同様に，定数個のホストにメッセージを送信するのみでるので $O(1)$ である．

メッセージ遅延については，まず，部分木の $diameter$ を計測するために部分木の葉ホストから根ホストに到達するまでにバランスよくノードが配置されている場合 $O(\log_{d_{max}}(n))$，最悪の場合で $O(n)$ である．解候補の選択も同様に，葉ホストから根ホストに到達するまで最良で $O(\log_{d_{max}}(n))$，最悪の場合で $O(n)$ である．解候補の探索の場合は，集められた解候補を１つのホストに集めるのみであるので $O(1)$ である．木の再構築に関しても解となるホストにメッセージを送信するのみであるので $O(1)$ である．

4　提案アルゴリズムの性能評価

提案アルゴリズムの性能評価のため，本稿で対象としている木構築問題に対して，文献 [2] において提案されている静的に木を構築するアルゴリズムとの比較を行った．シミュレーションでは，文献 [2] の手法との $diameter$ の比と再構築のコストを測定した．さらに，提案アルゴリズムにおける初期マルチキャスト木の構築は，参加した順に木に追加していくという方法であるので，初期マルチキャスト木の構築方法が，$diameter$ の増加に大きく影響する可能性が考えられる．そこで，初期マルチキャスト木を文献 [2] の従来アルゴリズムによって構築し，その後提案手法で再構築した場合との比較を行った．

本稿ではランダムグラフに対して以下の状況を想定し，提案アルゴリズムがどの程度の性能であるかを評価した．

- セッションがある程度進行するとホストの参加・離脱はあまり頻繁には発生しないと考えられる．よって，そのような状況を想定し，セッションに参加しているホスト数に対して，参加・離脱するホスト数の割合が小さい場合についてシミュレーションを行う．具体的には，初期に参加しているホスト数を１００と仮定し，その状態から各タイムスロットごと，ランダムに最大５の参加または離脱ホストを選択し，参加または離脱させる．これを１００タイムスロット繰り返す．
- 上記の状況とは逆に，セッションの開始直後においてはホストの参加・離脱が比較的頻繁に発生すると考えられる．よって，そのような状況を想定し，セッションに参加しているホスト数に対して，参加・離脱するホスト数の割合が大きい場合についてシミュレーションを行う．具体的には，初期に参加しているホスト数を１００ノードと仮定し，その状態から各タイムスロットごと，ランダムに最大３０の参加または離脱ホストを選択し，参加または離脱させる．これを１００タイムスロット繰り返す．
- 先程述べたように，初期マルチキャスト木を従来アルゴリズムで構築した場合に対して，初期に参加しているホスト数を１００と仮定し，その状態から各タイムスロットごと，ランダムに最大１０の参加または離脱ホストを選択し，参加または離脱させる．これを５０タイムスロット繰り返す．

以上の条件の元で，各タイムスロットあたりの $diameter$，および，各タイムスロットあたりの木の再構築による切り替えたリンク数の比較を行った．

以下に，ランダムグラフに対するネットワークの環境設定とシミュレーション結果を示す．

4.1　シミュレーション結果

4.1.1　ネットワークの設定

ランダムグラフのトポロジーに対するネットワークの設定は以下のようにした．

- ホストの配置については，３０００×３０００の座標空間にランダムに配置した．
- 各ホストの次数制約については，各ホスト一律に４とした．

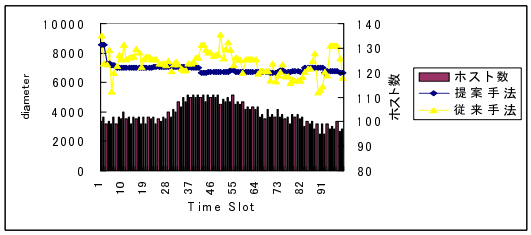

図 6: diameter の変化

表 1: 各タイムスロットあたりの平均

	$diameter$	切り替え (本)	時間 (秒)
従来手法	7310	41.6	0.61
提案手法	6897	1.6	0.14

- ホスト間のリンクコストについては，２ホスト間のユークリッド距離とした．

4.1.2　参加・離脱の割合が小さい場合の性能

図 6 にシミュレーション結果を示す．また，表 1 に各タイムスロットあたりの平均の $diameter$，リンクの切り替え数，および，計算時間を示す．これより，従来アルゴリズムと比較して提案アルゴリズムでは $diameter$ に関しては平均すると 6%程度良好な解が得られる結果になった．また，図 6 より，提案アルゴリズムは $diameter$ の変動が非常に小さいということがわかった．これは，提案アルゴリズムにおいて，新たなホストを追加する場合も $diameter$ の増加を最小限に抑え，また，離脱する場合も $diameter$ の増加を抑えて効率的に再構築が行うことが可能であるということがわかる．また．再構築におけるリンクの切り替えは従来アルゴリズムに対して 95%以上削減できる結果が得られた．静的に木を構築した場合では，ホスト数の変化が小さい場合でも最適な木を得ようとすると切り替えるリンク数は非常に多くなってしまうが，提案アルゴリズムではそれを最小限にした木の再構築が可能で，かつ，$diameter$ もわずかながらも良好な木が得られている．よって，提案アルゴリズムはこのような場合に対して非常に効率的であると言える．

4.1.3　参加・離脱の割合が大きい場合の性能

図 7 にシミュレーション結果を示す．また，表 2 に各タイムスロットあたりの平均の $diameter$，リンクの切り替え数，および，計算時間を示す．これより，従来アルゴリズムと比較して提案アルゴリズムでは $diameter$ の値は平均すると 15%程度増加するという結果が得られた．これは，全体のホスト数に対して参加・離脱の割合が大きい場合は，ホスト数の急激な変化がネットワーク全体に影響を及ぼすためであると考えられる．提案アルゴリズムでは構築された木の形状をできるだけ維持するため，そのような場合には静的に構築しなおした場合の方がよくなると考えられる．一方，再構築におけるリンクの切り替えに関しては従来アルゴリズムの 74%程度削減できるという結果が得られた．以上により，$diameter$ に関しては 15%程度の増加に抑えつつ再構築のコストを 74%減少させていることから，提案アルゴリズムは十分効率的に木の再構築が行うことが可能であることがわかる．

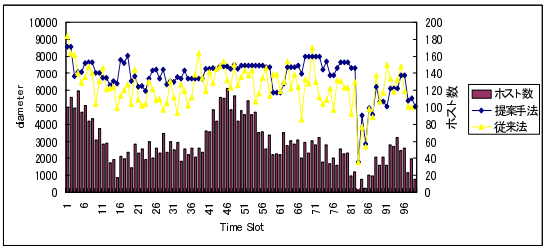

図 7: diameter の変化

表 2: 各タイムスロットあたりの平均

	diameter	切り替え (本)	時間 (秒)
従来手法	6109	42.0	1.1
提案手法	7021	11.0	0.21

4.1.4 初期マルチキャスト木構築において従来アルゴリズムを用いた場合との比較

図 8 にシミュレーション結果を示す．これより，提案アルゴリズムの初期マルチキャスト木は従来アルゴリズムを用いて構築した場合より，*diameter* は 30%程度の増加になっている．しかし参加・離脱を繰り返していくとすぐに同程度の解に収束し，さらに繰り返していくと，従来アルゴリズムを用いて初期マルチキャスト木を構築する場合よりもよい解に収束することがわかった．よって，提案アルゴリズムのように順次ホストを追加して初期マルチキャスト木を構築する方が，*diameter* の観点からも初期マルチキャスト木の構築コストの面からも有効であると考えられる．

5 まとめ

本稿では，オーバレイネットワーク上でのマルチキャスト木動的再構築アルゴリズムの提案を行った．提案アルゴリズムでは，ホストの離脱によりマルチキャスト木が複数の部分木に分割された場合，それらの部分木間を接続するリンクを限定することにより，平均で $O(\log n)$，最悪の場合で $O(n)$ の計算量で効率的に木の再構築を行う．シミュレーションの結果，従来の静的にマルチキャスト木を構築するアルゴリズム (計算量は $O(n^3)$) に対して，ランダムグラフ上で参加・離脱の頻度が大きい場合には平均 15%程度の *diameter* の増加に抑えつつ，再構築に要するコストを平均 74%削減することができた．また，その頻度が小さい場合には 6%程度 *diameter* を小さくでき，再構築に要するコストを 95%以上削減することができた．さらに，提案手法における初期マルチキャスト木構築が，その後の再構築の繰り返しによってどの程度 *diameter* に影響を与えるかを調べた．その結果，数回の再構築で従来アルゴリズムで初期マルチキャスト木を構築した場合よりもよい解に収束することがわかった．これらの結果から，提案アルゴリズムの有効性が確認できた．

今後の課題としては，より実際のネットワーク環境に近い環境を想定したシミュレーションが考えられる．分散環境では，ホスト間の通信に伴うメッセージ遅延に関しても考慮することが重要である．そのため，実環境におけるそのメッセージ遅延がどの程度アルゴリズムの性能に影響を与えるかを分析し，アルゴリズムの改良を行う必要があると考えられる．

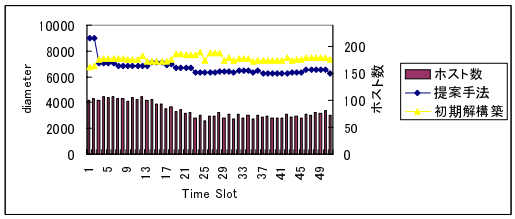

図 8: diameter の変化

参考文献

[1] J. Ho, D. Lee, C. Chang and C. Wong, *"Minimum Diameter Spanning Trees and Related Problems"*, SIAM J. Computing, Vol. 20, No. 5, pp.987–997, 1991.

[2] S. Shi, J. Turner and M. Waldvogel, *"Dimensioning Server Access Bandwidth and Multicast Routing in Overlay Networks"*, Proc. of Network and Operating System Support for Digital Audio and Video 2001, 2001.

[3] S. Shi and J. Turner, *"Routing in Overlay Multicast Networks"*, Proc. of IEEE INFOCOM 2002, 2002.

[4] R. Cohen and G. Kaempfer, *"A Unicast-based Approach for Streaming Multicast"*, Proc. of IEEE INFOCOM 2001, 2001.

[5] Y.-H. Chu, S. G. Rao and H. Zhang, *"The Case for Resilient Overlay Networks"*, Proc. of 8th Annual Workshop on Hot Topics in Operating Systems (hotOS-VIII), 2001.

[6] S. Ratnasamy, M. Handley, R. Karp and S. Shenker, *"Application-level Multicast using Content-Addressable Networks"*, Proc. of 3rd International Workshop on Networked Group Communication (NGC), 2001.

[7] S. Banerjee, C. Kommareddy, K. Kar, B. Bhattacharjee and S. Khuller, *"Construction of an Efficient Overlay Multicast Infrastructure for Real-time Applications"*, Proc. of IEEE INFOCOM, 2003.

[8] D. Pendarakis, S. Shi, D. Verma and M. Waldvogel, *"ALMI: An Application Level Multicast Infrastructure"*, Proc. of 3rd Usenix Symp. on Internet Technologies and Systems, 2001.

[9] P. Francis, *"Yoid: Extending the Multicast Internet Architecture"*, white paper, 1999. http://www.aciri.org/yoid/.

[10] D. Xu, S. Hambrusch and B. Bhargava, *"On Peer-to-Peer Media Streaming"*, Proc. of 22nd International Conference on Distributed Computing Systems (ICSDS), 2001.

[11] A. Chakrabarti and G. Manimaran, *"A Case for Scalable Multicast Tree Migration"*, Proc. of IEEE Globecom, 2001.

[12] E. M. Royer and C. E. Perkins, *"Multicast Ad hoc On-Demand Distance Vector (MAODV) Routing"*, IETF Internet Draft, draft-ietf-manet-maodv-00.txt, 2000.