

QoS マルチキャスト配送木の構築方式

西丸泰之¹

佐藤文明²

TV 会議システムなどの多人数が同時に通信するようなマルチメディアサービスの実現において、マルチキャスト通信が注目されている。マルチキャストは、複数の受信者に対して効率よく通信を行なうことができる。しかし、マルチメディア通信に必要である QoS をマルチキャストに導入する研究は、まだ多くの課題が残されている。本研究では、QoS を考慮したマルチキャストツリーを生成する新しい分散型の方式を提案する。この方式では、QoS パラメータを制御パケットに含めると共に、なるべく経路が共有されるためのアルゴリズムが含まれている。この方式をシミュレーションによって評価し、従来の集中型のアルゴリズムと同等程度の特徴をもつツリーが低コストで構築できることを示した。

An Algorithm to Construct the QoS Multicast Tree

Yasuyuki Saimaru¹

Fumiaki Sato²

Multicast communication attracts attention in the development of the multimedia services that many users communicate in the same time such as the TV conference system. Multicast is useful to communicate among many users in efficient. However, many issues are left in the QoS mechanism of multicast communication. In this paper, we propose a new decentralized mechanism to set up the multicast tree considering the multiple QoS. This mechanism includes the algorithm that many links in the tree would be shared, as well as the algorithm handling the QoS parameters. The simulation is used for the evaluation of the algorithm and the result shows that the tree is similar or better than the previous centralized algorithm.

1 はじめに

近年、インターネットの急速な普及に伴い、ネットワーク上を駆け巡るマルチメディアサービスの利用も多様化してきている。その中でも、リアルタイムで行われる電子会議システムや、映像配信を代表とした、多人数が利用するサービスが今注目されている。このようなサービスは、ネットワークを基盤として、多人数に、しかも大容量のデータを送信しなければならない。ネットワークの効率的利用の観点から、これらのサービスはマルチキャスト通信が適している。一方、このようなマルチメディアサービスにおいては、データ配信の際、帯域を効率的に利用し、通信コストを少なく抑える必要がある。さらに、リアルタイム性を重視しなければならないため、サービスの送信者から受信者までのネットワーク上の通信遅延時間も最小限にしなければならない。

マルチキャスト [1],[2] の経路は木構造となるた

め、一般にマルチキャストツリーと呼ばれる。これらの QoS (Quality of Service) を考慮したマルチキャストツリーを構築するアルゴリズムはいくつか提案されている [3],[4] が、まだ数は多くの課題が残されている。そこで本研究では従来のフラッド方式によるマルチキャストツリー構築アルゴリズムを拡張し、QoS を考慮した新しいマルチキャストツリーアルゴリズムを提案する。そして、従来研究として、Dijkstra のアルゴリズムを拡張した構築アルゴリズムによって生成されるマルチキャストツリーと提案アルゴリズムによって生成されるマルチキャストツリーの QoS としての性能を比較することにより、その評価を行う。

2 研究背景

QoS を考慮したマルチキャストツリーを構築するアルゴリズムはまだ少ない。そのような中で、Dijkstra のアルゴリズムを拡張した方式で複数の QoS を考慮したマルチキャストツリーアルゴリズムが提案されている [4]。本研究ではこのアルゴリズムを比較対照とする。本章ではこの従来研究のアルゴリ

¹ 静岡大学情報学部

Graduate School of Information, Shizuoka University

² 静岡大学情報学部

Faculty of Information, Shizuoka University

ズムを簡単に説明する。

2.1 Dijkstra のアルゴリズムの拡張

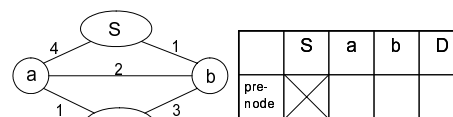
Dijkstra のアルゴリズムは送信ノードから順に目的関数（今回の場合コスト、遅延時間）の値の小さい経路をたどって最短経路を求める。目的関数の値の一番小さい経路以外の経路情報は破棄される。今回はその指標に対して必ずしも最適でなくてもよいため、最短経路を求める過程で得られる候補経路もスタックに保存しておく。

例として、このアルゴリズムを図1のネットワークで送信ノードを S とした場合の経路生成過程を示す。リンクに付加された数字が、目的関数とする QoS 指標を表すものとする。また、ネットワークの右側には前ノードテーブルが示されている。最上段が注目ノードであり、次の段以降、注目ノードに到達するまでの目的関数の値と直前ノードが表されている。

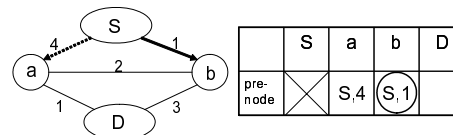
(1) まず、送信ノード S に接続されているノード a, b について、前ノードテーブルに S が書き込まれる。ここで、目的関数を計算すると、最小で到達できるノードは b であるため、b への経路が確定する（丸付き文字で表す）。

(2) 次に経路の確定したノード b を経由して到達できるノード a, D に注目すると、D にはこれまで前ノード情報がないため、前ノードテーブルに b が書き込まれる。一方、a には既に S という前ノード情報があるが、経路 $S \rightarrow a$ に要する目的関数値 4 に対して、経路 $S \rightarrow b \rightarrow a$ に要する値 3 のほうが小さいため、前ノード情報を書き換える。ここで、Dijkstra のアルゴリズムではこれまでの前ノード情報は破棄されるが、拡張されたアルゴリズムではこれをスタックに残す。ここで、経路 $S \rightarrow b \rightarrow D$ による目的関数値も 4 となり、経路 $S \rightarrow b \rightarrow a$ による目的関数値 3 を超えるため、ノード a についても経路が確定する。

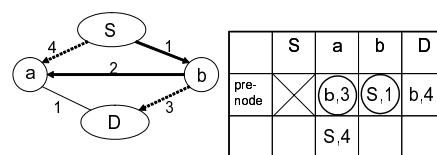
(3) 残った未確定ノード D に対して、前ノードテーブル上にある経路 $S \rightarrow b \rightarrow D$ と先に確定したノード a を経由する経路 $S \rightarrow b \rightarrow a \rightarrow D$ が比較される。ここでは、指標は等しい値となるため、これも前ノードテーブルに加えられる。



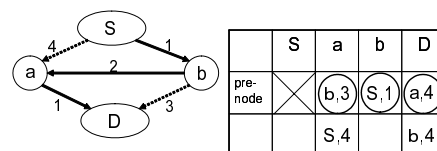
初期状態



(1)



(2)



(3)

図 1: 拡張された Dijkstra アルゴリズムの動作例

2.2 マルチキャストツリー生成

上記で述べた Dijkstra のアルゴリズムによって生成された前ノードテーブルの組み合わせによってできるルートのうち、要求された QoS をすべて満たすものを候補経路とし、さらにその候補経路を効率的に組み合わせることによりコストを抑える。要求された QoS とは、送信ノードから受信ノードまでのルートにおいて、そのルート中の全リンクにかかる総コスト、総遅延時間の許容できる限界値であり、この値以上の総コスト、総遅延時間がかかるルートは採用されない。

具体的なアルゴリズムは次のとおりである。ここでの C は各経路のコストであり、m は経路中に含まれる受信ノードのうちで、まだマルチキャストツリーに含まれていないものの数である。

(1) ある受信ノードに対して、そのノードを含む経路が 1 つしかない場合は、その経路を採用する。このような経路が存在しない場合には、何も行わずに、手順 (2) へ進む。

(2) 候補経路の中から C/m が最小のものを採用する。ただし、既に採用された経路に含まれている受信ノードはこの数に含めない。

(3) 手順 (2) で選択した経路上のリンクのコストを 0 とし、残りの候補経路のコストを再計算する。

(4) 手順 (2), (3) を全ての受信ノードがマルチキャストツリーに含まれるまで続ける。

上記の (2) の手順の中にある、受信ノード数 m というパラメータによって、なるべく多くの受信者で共有される経路が選択される。このことから、なるべく総コストの小さい木が構築されることになる。

2.3 問題点

この従来研究のマルチキャストツリー構築アルゴリズムは、送信ノードがネットワーク上の全てのリンク情報を保持し、それにより自らがルーティングの計算を行うというものである。よって、ネットワークが大きくなると送信ノードの保持する情報量も増え、マルチキャストツリー構築のための計算量も多くなる。これは送信ノードにかかる負担が大きくなることを意味する。そこで、本研究では送信者に負担をかけない全く異なった方式でマルチキャストツリーを構築するアルゴリズムを提案する。

3 提案する QoS を考慮したマルチキャストツリー生成方式

これまで本研究の研究背景を見てきたが、本章ではこの背景を基にして従来方式のマルチキャストツリー生成方式と、全く異なった方式で同様に複数の QoS を考慮したマルチキャストツリーを生成するアルゴリズムの仕組みについて説明する。

3.1 提案アルゴリズムの方針

本研究で提案するアルゴリズムは従来方式と同様、考慮する QoS をコストと遅延時間とする。ネットワーク上の各リンクにかかるコストや遅延時間を基に、マルチキャストのルートを決めていく。その際、最小限のコスト、また最小限の遅延時間で送信できるようなルートを選ぶ。これにより、生成されたいくつかのマルチキャストルートにおける、マル

チキャストの送信者から受信者までの 1 本のルートにかかる平均のコスト、遅延時間（平均コスト、平均遅延時間）を小さく抑えることができる。また、完成したマルチキャストツリーに使用される全リンクにかかるコストの総和（総コスト）も同様に可能な限り少なく抑えることができる。

また、別の QoS としてネットワークの帯域効率にも注目する。単純にコストや遅延時間を少なく抑えるだけでは、マルチキャストの、「複数のノードに送信する場合でも、経路を共有すれば、分岐点で複製することによりその同一経路に同じデータを 2 回以上流さなくて済む」という特性 [5] を生かすことはできない。そこで、マルチキャスト送信者から各受信者までのそれぞれのルートを生成する際、なるべくコストや遅延時間の小さいリンクを選択させる。これにより、コストや遅延時間の小さいリンクは複数のルートで採用される可能性が大きくなる。こうしてマルチキャストルートの中に共有されるリンクを多くし、帯域効率の良いマルチキャストツリーを作る。

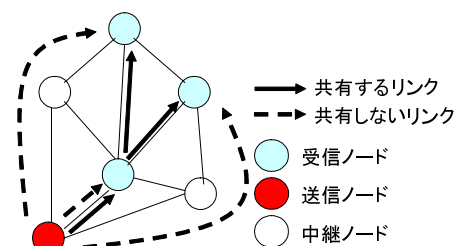


図 2: リンクの共有

3.2 提案アルゴリズム

前述した方針に基づいて、QoS を考慮したマルチキャストツリーを生成するアルゴリズムを提案する。本研究の提案アルゴリズムは従来から存在するフラッディング形式のルーティングアルゴリズムをベースとして、それを方針にあわせて拡張する形をとる。

3.2.1 マルチキャスト送信者

マルチキャストのルートを決める際、最初に行うのはマルチキャスト送信者がネットワーク上の各ノードに対して、そのマルチキャストに受信者として参加するかどうか問い合わせることである。この問い合わせはマルチキャスト参加告知パケット（以

下, Join Query と呼ぶ) をネットワーク上の全ノードに配信することにより実現する. 受け取ったノードがマルチキャストに参加するようであれば, その旨を送信者に知らせる. 本研究で提案するアルゴリズムは Join Query を全ノードに配信する過程で最適なルートを決定していく.

通過ノード名(Node ID[])
総コスト(Sum_Cost)
総遅延時間(Sum_Delay)
通過ノード数(N)

図 3: Join Query の構成

3.2.2 ノードの処理

Join Query を受け取ったノードは直前に通過したリンクの情報(コスト, 遅延時間)を基に, その Join Query の値を更新する. そして次にその更新された Join Query の情報を基に今度は, QoS 評価値 D という値を計算する. この D とは,

$$D = \text{Sum_Cost} + \text{Sum_Delay} + \frac{\text{Sum_Cost} + \text{Sum_Delay}}{N} \times p$$

という計算で求まる. これはつまり, 総コストと総遅延時間の和+一本のリンクにかかる平均のコストと遅延の和×p(可変定数)ということになる. 初めて Join Query を受け取ったノードはこの値を QoS 評価値として無条件で保存する. また, Pre_node としてこの Join Query を転送してきた前ノード名も保存する. ノードが保持するデータはこの2つである. そしてこの Join Query を, このパケットが通過していないノード(Node ID にないノード)へと転送する.

その後, このノードにまた別のルートを通ってきた Join Query が到着した場合, この Join Query についても同様に D の計算を行う. そしてここで計算された D の値が, 既にノードに保存されている QoS 評価値よりも大きかった場合, その Join Query はこの時点で破棄され, このノードから別のノードへ転送されることはない. また, 要求された QoS として, 送信ノードから受信ノードまでのルートにおいて, そのルート中の全リンクにかかる総コスト, 総遅延時間の許容できる限界値を設け, この値以上の総コスト, 総遅延時間がかかるルートを通じてきた Join Query もこの時点で破棄させる. もし, 計算された D の値が, 既にノードに保存され

ている QoS 評価値よりも小さかった場合は, 初めて Join Query を受け取った時と同様の処理を行う. つまり, ノードは QoS 評価値をこの D の値に更新し, この Join Query を転送してきた前ノード名を Pre_node にセットする. そしてこの Join Query を, このパケットが通過していないノード(Node ID にないノード)へと転送する.

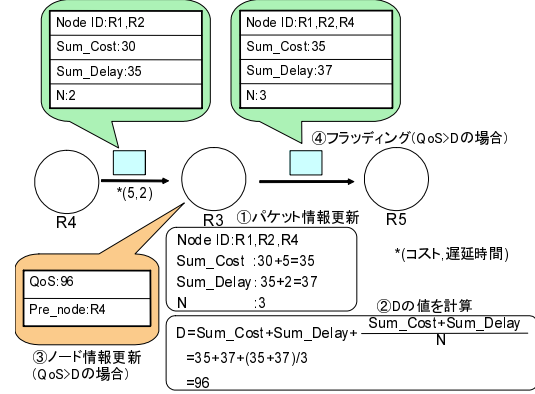


図 4: ノードの処理

3.2.3 マルチキャスト参加ノード

もし, Join Query を受け取ったノードがマルチキャスト参加ノードであった場合, そのノードは上記の処理に加え, ルートを決定する処理を行う. これは, このマルチキャスト参加ノードが, 一番最後にこのノード自身が保持する QoS 評価値を更新した Join Query の通過してきたルートを逆にたどり, マルチキャスト送信ノードにそのルートを知らせることによって実現する. 実際の処理は次のとおりである.

単純に参加ノードから, マルチキャスト参加応答パケット(以下, Ack と呼ぶ)を, マルチキャスト送信ノードに向けて送信する. Ack はこの参加ノードをスタート地点として, マルチキャスト送信ノードまで各ノードに記録されている Pre_node のノードをたどる. Ack はそのたどっていく過程で通過したノード名を順に記憶していく. マルチキャスト送信ノードはその Ack が保持する最終的な情報を受け取ることで, マルチキャスト送信者からこの参加者までのルートを把握することができる. この処理をマルチキャスト全参加者が行うことで, ルートが集合し, マルチキャストツリーが完成する.

4 シミュレーション

本章では、前章で説明したアルゴリズムによってできたマルチキャストツリーの性能を、従来研究のアルゴリズムによるマルチキャストツリーの性能と比較することによって評価する。評価はシミュレーションによって行った。

4.1 シミュレーション環境

ネットワークモデルとして、ノード数が400で、各ノードに繋がるリンク数平均3のランダムグラフを生成し、各リンクに遅延時間とコストをほぼ同じ重み付けになるような範囲の一樣乱数としてそれぞれ付加したものを、100通り用意した。このグラフの各ノードの中から受信ノードを20～200個選出した。また、提案アルゴリズムのDを求める計算式にある、パラメータ p の値を変化させてデータを取得した。

4.2 シミュレーション方法

上記の環境で、従来方式と提案方式による2通りのマルチキャストツリーを作成した。この2通りのマルチキャストツリーを比較する項目として、総コスト、最大遅延時間を挙げた。ここでの総コストとは、完成したマルチキャストツリーに使用される全リンクにおけるコストの総和であり、最大遅延時間とはマルチキャストの送信者から、受信者までのルートにかかるリンクの遅延時間のうち、最大のものである。総コストから帯域の効率の度合いも確認できる。

4.3 シミュレーション結果

総コスト、最大遅延時間に関して、それぞれ100通りのネットワークモデルの平均値を基にグラフを作成した(図5, 図6)。また、Dの計算式中にある可変定数 p の値と総コストの関係を示す(図7)。

この結果から、遅延時間に関して従来方式よりも減少したことが確認できた。しかし、総コストに関しては増加がみられた。また、 p の値が増加するに従い、総コストの値が減少していることがわかる。

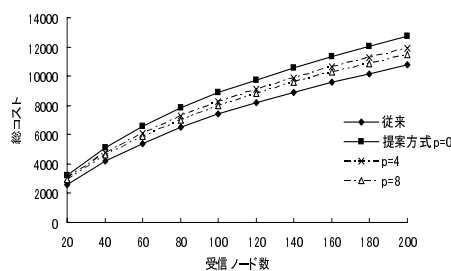


図 5: 総コスト

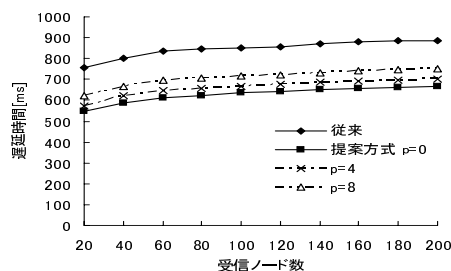


図 6: 最大遅延時間

4.4 考察

この結果から、遅延時間に関して従来方式よりも減少し、一方コストに関しては増加がみられた。従来方式はコストと遅延時間両方をQoSとして考慮したもの、遅延時間は候補経路にのみ考慮されていて、最終的なルート決定はコストに依存していた。それに対し、提案方式はコストと遅延時間の重みを同一と考え、ルートを決定するように設計されている。この理由から、従来方式よりコストの制限を緩和した結果、その分だけ遅延時間の抑制ができたと考えられる。

また、 p の値が増加するに従い、総コストの値が減少していることがわかった。QoSを考慮する判断基準となるDの式が総遅延時間と総コストの和という要素と、一本のリンクに掛かるコストと遅延時間の少なさという2つの要素からなっているが、その中で、 p というパラメータはこの2つの要素のうち、どちらの要素を重要視するか重み付けをする役割を果たす。 p の値が大きければ大きいほど、総遅延時間や総コストが多少大きくても、できる限り一本のリンクのコストや遅延時間が短いルートを選択し、帯域効率のよいルートに集中しようとする。逆に p の値が小さければ小さいほど、一本のリンクにルートを集中させることは抑え、最短コスト、最短遅延時間でのルートを選択しようとする。これは、

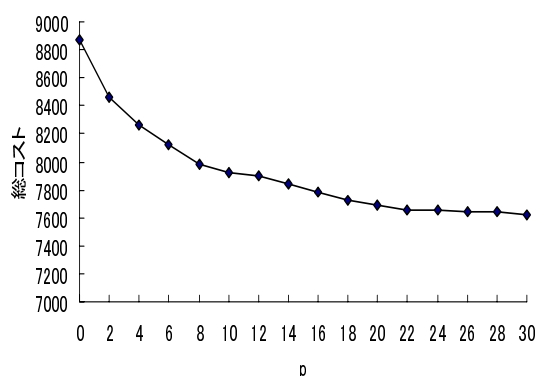


図 7: p と総コストの関係

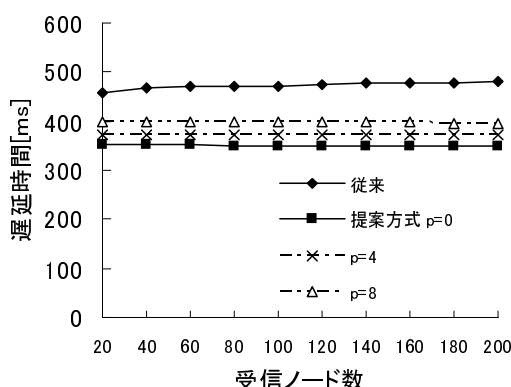


図 8: 平均遅延時間

平均遅延時間と p の関係 (図 8) においても推測できる。ここでの平均遅延時間とは、マルチキャスト送信ノードから 1 つの参加ノードまでのルートにおける各リンクの遅延時間の和を参加ノード全てに対して算出し、それを全て足したものを参加者数で割ったものである。 p の値が増加すれば増加するほど平均遅延時間が増加している。 p の値を増加させれば、帯域効率のよいルートを選択しようとし、その副作用としてマルチキャストにおけるそれぞれのルートの遅延時間は増加する。

5 おわりに

本研究では、従来のフラッディング方式のマルチキャストツリー構築アルゴリズムを改良し、作成過程に複数の QoS の要素 (コスト, 遅延時間, 帯域効率) を導入し、QoS を考慮したマルチキャストツリー構築アルゴリズムを提案した。そしてこのアルゴリズムによって生成されたマルチキャストツリーをシミュレーションによって評価した。このシミュ

レーション結果により、このアルゴリズムでコストと遅延時間を抑制することが可能であることがわかった。また、QoS を考慮する評価値 D における、 p のパラメータの値により、どの程度リンクを集中させるかを制御できることを確認した。

今後の課題として、ルート決定の際に、候補ルートの中に存在するマルチキャスト参加ノード数をパラメータとして加え、より帯域効率の良いマルチキャストツリーになるように改良することが考えられる。また、現在のアルゴリズムではノードは Join Query を D の更新回数分フラッドするが、これも帯域の浪費につながる。ノードのフラッド回数を抑制する仕組みを追加したい。また、本研究のアルゴリズムではマルチキャストツリーができた後、その参加や脱退の処理に関しては考慮されていない。これも検討したい。そして、様々なネットワークモデルに対して、 p の最適値が確認できていないのでこれを検討することも挙げられる。

参考文献

- [1] D.Waitzman, C.Partridge, S.Deering, “ Distance Vector Multicast Routing Protocol ” RFC 1075, Network Working Group, 1988
- [2] D.Estrin, et.al, “ Protocol Independent Multicast - Sparse Mode (PIN-SM) ” Protocol Specification, RFC 2362, Network Working Group, 1997
- [3] 関口隆昭, 藤川賢治, 岡部寿男, 岩間一雄, “ QoS 保証マルチキャストプロトコル SRSVP における階層化 PathQoS ” 情報処理学会マルチメディア通信と分散処理研究会研究報告 102-28, 2001 年 3 月 22 日
- [4] 木下 和彦, 神原 正義, 谷岡 秀昭, 村上 孝三, “ 複数の QoS 要求を満たすマルチキャストツリーを生成するリンク状態アルゴリズム ” 電子情報通信学会論文誌 B Vol.J85-B No.10 pp.1738-1747, 2002 年 10 月
- [5] H.Tode, Y.Sakai, M.Yamamoto, H.Okada, and Y.Tezuka, “ Multicast Routing Algorithm for Nodal Load Balancing ” Proc.IEEE INFOCOM '92, pp.2086-2095, New York, May 1992.