多候補送信の再帰的ルーティングによる安定した端末型マルチキャスト

清水周一[†1]

本論文では，不安定な系においても安定してデータ配信を行うための端末型マルチキャストの方法について示す．特徴は，配信のための経路を固定せずに，送信先の状況に応じて動的に宛先を変更すること，さらに，安定した状況であっても頻繁に宛先を変更することである．この流動的な経路構成により，性能の低いノードや障害のあるノードは配信経路上の下流に追いやられて，全体として安定した配信を実現する．また，各ノードに経路の決定を任せてもループを生じない分散型の方式を可能とすること，エラー処理は送信側主導で局所的に行うことなどにより，配信の規模に対してスケーラビリティに優れる．流動的経路構成は，トラフィックの平均化および高速化にも寄与する．本論文では，プロトコルの詳細に基づき，遅延に関する性能を解析する．遅延時間では，固定配信経路の方式に比べると約 1/2 倍に短縮されることを示す．また，喪失符号の導入とエラー率などに基づく配信システムの設計についても解析的に示す．

Recursive Point-to-group Routing for Reliable End-host Multicast

Shuichi Shimizu[†1]

This paper presents a novel application-layer multicast technique, *recursive point-to-group routing*, in which data delivery paths are not fixed but change dynamically and frequently even when no failures occur at receiver hosts. The advantages of our technique include (1) automatic and rapid adaptation of delivery routes to the non-uniformly and temporary fluctuations in host performance, (2) well-distributed and well-balanced network-relaying traffic for all hosts over the delivery system, and (3) scalability based on sender-based path determination and local error recovery. Each node locally determines next forwarder, but no loop occurs due to exploiting hierarchical network structure. In this paper, we present details of protocol and show analytical results of delivery delay time by using a queuing model. With low system utilization, the delivery time is about 1/2 of one in the conventional path-fixed approaches. We also show how to introduce erasure code and design system parameters to control very small error rate.

1. はじめに

マルチキャストは，多数の宛先にデータを同時配信するための効率的なインフラストラクチャであり，たとえば，多数視聴者向けのライブ・ストリーミングやバルク・データの同時配布など，その用途は広い．ここでのポイントは，ネットワーク資源の使用効率を上げるため，データが重複して同じ経路を通ることのないように適切な配信経路を構成することである．ネットワーク層のマルチキャスト（IP Multicast）[10] では，配信元から多数の宛先に向けて，ルータ機器をノードとした木構造状の配信経路を構築する．この方法は，トラフィックなどの観点で最適な配信経路を実現することができる一方で，広範囲のセキュリティ管理やフロー制御などの問題が未解決のまま残っているため，広く運用されるには至っていない．

アプリケーション層マルチキャスト（あるいは端末型マルチキャスト）は，ルータ機器の代わりにエンド・ホスト（端末）をルーティング（中継）ノードとしたオーバレイ・ネットワークの上に，ユニキャストに基づく配信経路を構築する．そして，この技術の成熟したユニキャストを使って，ネットワーク層マルチキャストにおける問題点を解消してきた[1),2),22)–24),29),30)]．また，配信の性能に関してネットワーク層マルチキャストに近づけるべく，多くの議論が行われてきた[9),13),21),26)]．しかし，一方で，オーバレイ・ネットワークに起因する性能の変化や不安定性など新たな問題[3),7),8),15),28)]もある．

本論文では，端末ノードの性能が劣化したり，予測

†1 日本 IBM 株式会社東京基礎研究所
Tokyo Research Laboratory, IBM Japan

しない障害のために切り離されたりするといった不安定なインフラでも，その上に構成したオーバレイ・ネットワークにおいて，安定したデータ配信を継続するための方式について示す．その本質的な特徴は，システムのダイナミクス（力学的原理）を利用すること，つまり，データ配信のための経路を固定せず，転送先の状況に応じて動的に，かつ頻繁に経路を変化させること[27]にある．これにより，不安定であったり性能が低かったりする端末ノードは回避されて，配信経路の下流に位置する頻度が高くなり，他のノードのデータ受信に影響を与えにくくなる．また，経路の決定を中央集権的ではなくて，各ノードに任せる分散型の方式によりスケーラビリティを保つ．

以下では，本手法の基本アイデアとプロトコルの詳細を示した後，遅延時間に関する解析を行い，最後に喪失符号を導入した際の誤り率に基づくシステム設計と安定性について議論する．

2. 不安定な系における端末型マルチキャスト

1つの配信サーバ（ルート）から多数の受信クライアント（ノード）に向けて，同一のデータを配布するための配信ネットワークを考える．大きなデータは複数の小さなセグメント（パケット）に分割されてから，順に受信ノードに向けて送信されるとする．そして，各末端ノードで必要に応じて，元のデータに復元されるとする．

端末型マルチキャストでは，一部のノードにパケットを中継する処理を担わせて，ルートから各ノードに向けた配信経路を構成する．このとき，ネットワーク資源を有効活用して，配信の規模（ノード数 N）に対してスケーラビリティを確保するために，**図1**の上側に示すような木構造状の配信経路を構成して，1つの受信パケットを1つあるいは2つ以上の宛先ノードに中継転送する仕組みを作る．

端末型マルチキャストの中継ノードは，ルータといった専用のネットワーク機器ではなく，一般のPCなどのプロセッサを利用するので，計算やワープロ操作など本来のタスクを同時に並行して行っているなど，性能の面で不安定性の問題をはらむ．また，中継ノードの障害や離脱などによる下流ノードへの影響も重大な問題となる．このとき，代替の配信経路が作成されるまで，下流ノードへの中継転送処理が滞るので，リアルタイム性を必要とするライブ・ビデオなどの配信アプリケーションなどの場合には，サービスの品質を落とすことになる．このような問題を回避するためには，系の不安定性を前提とした，ロバストなデータ配信の仕組みが必要である．

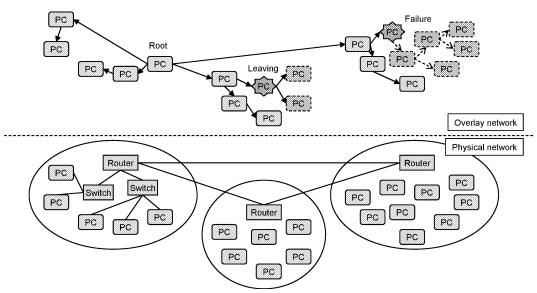

図1 オーバレイ・ネットワーク上の固定配信経路
Fig. 1 Path-fixed multicast on top of overlay network.

3. 多候補送信の再帰的ルーティング

安定したデータ配信を実現するためには，不安定なノードを動的に回避した経路の構築が重要である．以下では，その実現手法および特徴について示す．

3.1 ID に基づく階層的配信

図2に，ノードのIDに基づいた再帰的で階層的なルーティングを示す．この方法は，Minimum Broadcast Network（MBN）[12),25)]であり，1つのデータ・パケットを $O(\log_2 N)$ の時間で配信する性能を持つ．

例では，各ノードは4ビットのユニークなIDを持っているとし，ステージ0（S_0）において，配信元のルートからID = 0000のノードがパケットを受信するとする．その後，ステージ1（S_1）において，IDの1ビット目の異なるID = 1000のノードにパケットを転送する．ステージ2では，それぞれが，2ビット目の異なるノードに転送する．これを再帰的に，ステージ4まで繰り返すと，木構造状にすべてのノードへの配信が完了する．ここでは，説明の便宜上ステージを導入し

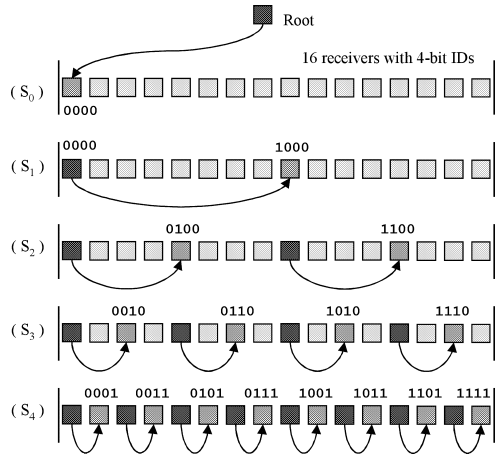

図2 IDに基づく再帰的ルーティング
Fig. 2 Recursive routing based on node ID.

ているが，各ステージは必ずしも同期する必要はない．

なお，ID はネットワークの構造に合わせて割り当てると，徐々に局所に落ち着く効率の良い配信経路を構成することができる．IPv6[11] のアドレスは，この性質を持つ．IPv4 のアドレスを代用する場合には，特に下位ビットが必ずしも構造を反映しないこともあるが，この階層的配信の動作を妨げることはない．

3.2 多候補送信による安定経路構築

図 2 では，宛先ノードが固定された階層的な配信経路の構成方法を示したが，k 番目のステージにおいて，k 番目のビットが反転していて，かつ，1 から $k-1$ 番目のビットがすべて一致したノードの中から宛先を選んでも，同様の再帰的な判断により，階層的でループのない配信経路を構築することができる．この様子を**図 3** に示す．宛先グループの階層は，ノードによって異なる．たとえば，ノード 0000 から見ると，宛先グループは {1∗∗∗}，{01∗∗}，{001∗}，{0001} の 4 つであるが，ノード 0101 では，{1∗∗∗}，{00∗∗}，{011∗}，{0100} である．

多候補送信を実現するには，ノード自身の ID と，送信先アドレスに対応づけられた宛先ノード候補の ID を知る必要がある．この情報があれば，上記のルールに従って，そのノードからの宛先グループの階層を構築することができる．ここで，各ノードは，サービスに参加するときに ID に関する必要な情報を知らされるとする．なお，この方式を，単ノードから宛先グループへの転送ととらえて，point-to-group（P2G）ルーティングと呼ぶことにする．配信経路は，この P2G ルーティングを再帰的に行うことで構成される．

ID に基づく宛先グループの決定ルールを守る限り，多数の候補の中から各ノードが自身の宛先を選ぶという分散型の方法でも，必ず，すべてのノードに至る経路をループすることなく構築することができる（分散型の経路構築）．宛先候補の中から，パケットの転送先として安定したノードを選択すれば，安定したノードが優先的に配信経路の上流に位置する，逆に，不安定と判断されたノードが下流に追いやられる形の配信経路を構築することができる（安定した経路構築）．なお，宛先の候補については，すべてのノードを知る必要はなく，宛先グループあたりの最大数を設定すれば，各ノードが管理する情報も，全体の規模，すなわち，ノードの総数 $N = 2^n$ にほとんど影響を受けず，$O(\log_2 N)$ の情報量でスケーラブルとなる．

図 4 は，先の 2 つの配信経路を木構造で表現したものである．各ノードの順番が入れ替わり，経路が大幅に変化していることが分かる．また，ループは生じない．

なお，図 2 などには，すべての ID が揃った状況を示したが，一般的には，欠番が生じているので，ステージ数は $n = \lceil \log_2 N \rceil$ である．新規ノードが参加する際には，配信元のルートが ID に基づいてオーバレイ上の位置を決定して，他の参加ノードに新しい宛先候補として通知する．新規ノードはこのとき，自身の宛先候補を知らされるが，送信元ノードについては情報を持たない．このように，受信ノードが送信元に要求を出す従来の方式とは異なり，本方式は，送信ノードから一方的にデータを届ける配信モデルである．

ノードが離脱する際には，オーバレイ・ネットワークが一時的に分断されるが，パケットが行き先を失わないように，緩やかな離脱処理（後述）を行う．また，予期しない離脱が起きて，緩やかな離脱処理ができな

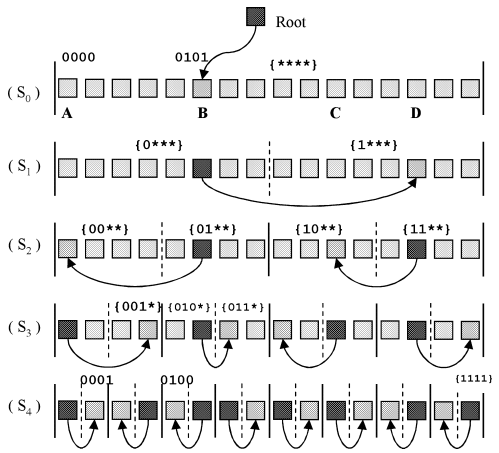

図 3 多候補に基づく point-to-group ルーティング
Fig. 3 Path-varying routing based on group unicast.

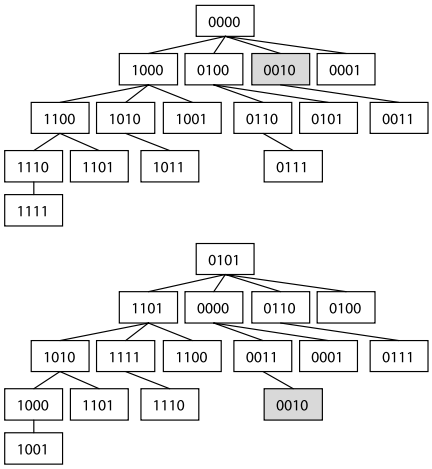

図 4 可変配信経路の構築
Fig. 4 Path-varying delivery paths.

かった場合には，送信側ノードによる再送処理を行い，パケットの喪失を防ぐ．詳細は，4.3 節に示す．

3.3 多候補送信による利点

以下では，再帰的 P2G ルーティングの特徴および利点について議論する．

3.3.1 トラフィックの平均化

宛先候補の中から次の安定した転送先を選択するとき，ラウンド・ロビン方式のように順に，あるいは，ランダムに選ぶことにより，配信経路を頻繁に変更することができる．これにより，配信経路上での各ノードの位置が変化するので，1 パケットの受信に対する転送のトラフィックを n から 0 の間で変化させることができる．各ノードは必ず配信元ルートからの送信レート λ と同じパケット量を重複なく受信する．一方，転送による送信については，平均的に $\lambda(1-1/2^n)$ の送信レートとなる．すなわち，n が十分に大きければ，λ-in/λ-out という各ノードでバランスのとれた中継処理が実現される．従来の m-分木の固定配信経路方式では，λ-in/$m\lambda$-out（$1 < m$）というアップリンクに負荷のかかるバランスなので，ADSL などの非対称バンド幅を特徴とするインフラにおいては，アップリンクの制限により性能を下げることになっていたが，このような問題も解消される．また，各ネットワーク階層（ステージ）においても，受信方向のダウンリンクは λ，転送方向のアップリンクは k 番目のステージで $\lambda(1-1/2^k)$ なので，配信の規模によらず，ネットワーク構造の上位でも同様にバランスのとれたネットワーク・トラフィックを実現することができる．

3.3.2 連続データ送信時のスループット増大

各ノードが単位時間に，送信処理あるいは受信処理のいずれか一方を行うと仮定すると，MBN は最短の配信時間を達成する[12),25)]．しかし，連続してデータを配信する場合には，固定の MBN を繰り返して利用するよりも，m-分木（$m \geq 1$）の方が配信時間が短くなることが示されている[14)]．これは，MBN では多くのノードがアイドル状態であるにもかかわらず，最上流のノードが送信処理で忙しく，複数の宛先に向かって送信し終えるまでは次のデータを受信できないためである．一方，1 分木（$m = 1$）では，いずれのノードも等しく送信と受信を繰り返して，均等に配信処理に貢献することができるため，ノード数に比べてデータ数が多い場合には有利となる．

再帰的 P2G ルーティングでは，送信可能な宛先を探しながら，つねに新しい MBN を動的に構成して配信を行うため，上流のノードで待たされることが少なくなる．したがって，1 分木と同等の効率を達成できるとは限らないが，上記のような固定の MBN で生じる配信遅延の問題は生じない．

ラウンドトリップタイム（RTT）の増加に応じてスループットが減少する[20)]ような TCP などを送信に利用する場合には，同一の宛先グループに対して複数（m）の接続でデータを送信すると，実質的にスループットが m 倍に増えるので，P2G ルーティングは，この点においても高速化にも大きく寄与する．

3.3.3 バースト誤りの回避

経路が頻繁に変化することにより，エラーを引き起こす状況が継続しないので，バースト的な喪失が起こりにくく，後述する喪失符号の復号においても有利である．

4. プロトコル

この章では，多候補送信の再帰的ルーティング・プロトコルについて疑似コードを用いて定式化する．

4.1 パケットの伝播

各ノードはパケットを受信すると，送信元のレベルに応じて，次のグループへとデータ転送を行う．そのために，各ノードは以下の受信用プロセスを持つ．

RECEIVER
1. Receive p from s,
2. If p is a DATA, do FORWARD(s,p),
3. If p is an ACK, do ACTIVATE(s),
4. If p is an INFO, do UPDATE(p),
5. Dispatch PROCESS(p),
6. ACKNOWLEDGE(s),
7. Goto step.1

ステップ 1 では，いずれかのノード s からパケット p を受信する．このとき，受信するデータがなければブロックして待つ．もし，受信したのがデータ・パケットならステップ 2 で転送処理（FORWARD）を行う．もし，ACK 型なら，送信元ノード s が次回の宛先として選択されるように活性化（ACTIVATE）する．もし，INFO 型なら，宛先に関する情報を更新（UPDATE）する．これら 3 つの詳細については後述する．ステップ 5 では，受信したパケット p について，たとえばストリーム処理などの本来の固有処理を非同期的に行う．ステップ 6 で，送信元 s に正常に処理したことを知らせるために ACK を返して，最初のステップに戻る．1 以外は，いずれのステップもブロックしない．

転送処理は，以下のとおりである．

FORWARD(s,p)
1. $k \longleftarrow$ LEVELOF(s),
2. For i from $k+1$ to n, do ENQUEUE(p,G_i).

ステップ 1 では，自分から見た送信元 s のレベル（ステージ）k を決定して，ステップ 2 で，その次のレベルから最後のレベル（$n = \lceil \log_2 N \rceil$）までの宛先グループ G_i に関連づけられた待ち行列（キュー）に，データ・パケット p を投入する．この投入処理はブロックしない．なお，送信元のレベルは，自身の ID と上位ビットから比較して，最初に不一致となるビット位置がレベルとなる．また，送信元のルートから受信したとき，送信元レベルはゼロである．

ここで，3.2 節で述べたように，宛先グループは自身の ID に基づいて決定される．グループ G_i には，ID の上位 $i-1$ ビットが一致して，i ビット目が反転して，残りのビットは任意である宛先ノードが含まれる．

各宛先グループ G_i には，次の非同期処理が関連づけられる．

WORKER(G_i)
1. $p \longleftarrow$ DEQUEUE(G_i),
2. $g \longleftarrow$ CHOOSE(G_i),
3. Dispatch SERVICE(p,g),
4. Go to step.1

ステップ 1 では，待ち行列から 1 つだけパケットを取り出す．空の場合には，先の FORWARD 処理により投入されるまで待つ．ステップ 2 では，グループの中から 1 つだけ宛先を選び出す．準備のできている（active）宛先ノードがなければ，いずれか 1 つが準備できるまで待つ．ステップ 3 では，パケット p を宛先 g に転送するためのサービスを非同期的に呼び出す．ここではブロックせず，ステップ 4 で処理の最初に戻る．

非同期の SERVICE プロセスでは，パケット p を宛先ノード g に送信する．

SERVICE(p,g)
1. DEACTIVATE(g),
2. SEND p to g.

まず，ステップ 1 では，先の CHOOSE で選択されないように，宛先ノード g に選択不可のマーキングをする．このマーキングは，先の RECEIVER プロセスの ACTIVATE 操作で解除される．次のステップ 2 で送信を完了したら，このプロセスを終える．

以上のように，各受信ノードでは，受信プロセス（RECEIVER）が 1 つ，待ち行列のプロセス（WORKER）が宛先グループの数だけ常駐し，また，転送用プロセス（SERVICE）が必要に応じて生成される．

4.2 新規ノードの参加に関する処理

新規のノードが参加するときは，ルートに参加要求を送る．ルートから初期情報として，宛先グループの構成とその宛先候補ノードを知らされるので，宛先グループなどに接続する準備が完了したら再度ルートに返答する．

ANNOUNCE
1. $p \longleftarrow$ NODEINFO,
2. For i from 1 to n, do ENQUEUE(p,G_i).

次に，新規ノードは自身の ID とアドレスを含む情報パケット（NODEINFO）を作成して，他のノードに通知する．情報の送信は，すべての宛先グループに投入することにより行う．投入後の処理は，先の WORKER プロセスが実行する．

この情報パケットを受信したノードでは，必要に応じて，宛先グループの送信先候補を更新する．

UPDATE(p)
1. g and $k \longleftarrow$ PARSE(p),
2. UPDATEGROUP(g,G_k).

情報パケットより宛先ノード候補 g とそのレベル k を解釈して，各宛先グループ G_k の更新を行う．ただし，グループ G_k に十分な数の宛先候補が登録されている場合や，すでに g が宛先として登録されている場合には何もしない．

4.3 ノードの離脱処理

ノードが離脱するときは，配信ネットワークの一部が分断されることになるので，パケットが喪失することがある．これを回避するために，離脱する場合には，まず，送信元のノードに切断することを通知する．切断の予告を受け取ったノードでは，自身の宛先候補からそのノードを抹消する．これにより，離脱するノードに次のパケットを送ることはない．離脱ノードは，次に，転送処理の未処理分を完了する．この一連の手続きにより，配信ネットワークの一部が切れてもパケットが行き場を失うことはないので，喪失を回避できる．

しかし，ノードに予期しない障害などが起きた場合には，このような緩やかな離脱の処理は必ずしも完了

しない．この状況を救うために，宛先ノード g の切断を検出した[⋆1]ノードでは，最後にそのノードに送ったパケットを 1 つ（あるいはそれ以上）履歴から取り出して（RETRIEVE），他の宛先候補に再送する．

RESEND(g)
1. $p \longleftarrow$ RETRIEVE(g),
2. $k \longleftarrow$ LEVELOF(g),
3. ENQUEUE(p,G_k).

この送信側主導のエラー回復処理は，受信側ノードが集中的に再送を問い合わせる方法と比べると，スケーラビリティを損なわずに迅速な回復処理を行うことができる．なお，パケットの再送により，同一パケットを多重に受信することもあるので，各ノードでは連番を利用して重複を排除する必要がある．宛先ノードの切断を検出できない場合でも，ACK が戻らなければ次の宛先としては選択されないので，次のパケットが失われることはない．なお，少数のパケット喪失は，後述の喪失符号で救うことができる．

4.4 ルート・ノードにおける処理

これまで示してきたように，配信の規模に対して，各ノードにおいてはスケーラブルな処理が実現されているが，一方，ルート・ノードでは全参加ノードを登録して管理する必要上，スケーラブルではない処理が必要となる．

まず，配信の開始点としての処理は，配信すべきパケット p それぞれについて以下のように処理する．

DELIVER(p)
1. $g \longleftarrow$ CHOOSE(S),
2. Dispatch SERVICE(p,g).

登録されたノード集合 S の中から 1 つ準備のできたノード g を選び，送信処理を非同期で開始する．ここで，集合 S のサイズは N である（$N = |S|$）．

新規ノード s を受け付けたときは，

ACCEPT(s)
1. $S \longleftarrow \{s\} \cup S$,
2. For i from 1 to n, setup G_i for s,
3. Notify s of $\{G_1, \ldots, G_n\}$.

それを登録するとともに，s から見た階層グループ $G_1, \ldots, G_n$ を構成して通知する．

ノード s の離脱を検出したときは，その登録を抹消するとともに，代替ノード g を選んでアナウンスする．ここで，代替ノードは，離脱したノードの ID を上位ビットから比較したとき，最も長く一致する既存ノードとする．

DROP(s)
1. $S \longleftarrow S - \{s\}$,
2. $g \longleftarrow$ FINDNEAR(s),
3. Tell g to announce itself.

代替ノードをアナウンスするのは，宛先候補の離脱のために十分な候補数を持たなくなったノードに新たな宛先候補を補充するためである．

5. 性能解析

本章では，待ち行列モデルを用いてプロトコル解析を行い，宛先を増やすことにより性能が向上することを示す．さらに，データ・パケットの到着が遅れて本来の処理に間に合わなかった場合をデータ喪失（ロス）ととらえて，それを回復するための喪失符号の導入，および冗長性などの設計パラメータや最終的な誤り率などについて導出方法を示す．最後に，安定性について考察する．

5.1 待ち行列モデル

各データ・パケットが配信元のルートから末端の受信ノードに届くまでには，最短で 1 ステップ（リンク），最長で $n+1$ ステップの配信経路を通る．本手法では配信経路は固定ではないので，どの受信ノードにおいても，データ到着にかかる時間はデータ・パケットによって変化する．また，各中継ノードは専用のルータ機器ではなく，他のプロセスなどが計算資源やネットワーク資源を並行して使用していることもあるので，次のノードに転送するまでの時間は一定とせずに，確率的に変化（分布）すると仮定する．以上の考察に基づき，本手法を解析するために，**図 5** に示すように，多段の待ち行列（キュー）モデルを導入する．

ここでは，問題を簡単にするために，全部で $N = 2^n$ の受信ノードがあるとして，0 から n ステージまでの $n+1$ 階層を考える．なお，2^n より受信ノードが少ない場合には，配信経路の平均長は短くなるので，この問題設定は最悪のケースを扱うことになる．なお，ス

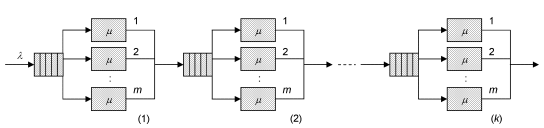

図 5 待ち行列モデルによる多段配信経路
Fig. 5 k-step path consisting of queuing systems.

⋆1 TCP で接続した場合などに可能

テージは階層を表し，一方，ステップはリンク数（距離）を表す．

受信ノードが配信元のルートから直接にデータ・パケットを受信する場合を 0 番目のステージと呼ぶことにする．各受信ノードは対等に扱われるので，これは確率的に $1/2^n$ で起こる．また，k 番目のステージ（$0 < k \leq n$）でデータを受信する確率は，$1/2^{n-k+1}$ で表される．なお，図 2 に示すように，k 番目のステージ（$0 \leq k < n$）でデータを受信したノードは，$k+1$ 番目以降のステージで中継処理を行う．一方，ノードがデータを受信するまでに k ステップの伝播が行われる確率は $\binom{m}{k}/2^n$ である．

図 5 は，k ステップ後にデータを受信するまでの様子をモデル化している．各ステップでは，m 候補の宛先のうちの 1 つを選んでデータを中継送信する．これを m サーバの待ち行列モデルで表現する．すなわち，サーバに空きがあればただちにデータを送り転送処理を開始するが，どれも忙しく空きがなければ，待ち行列に積んで空きができるまで待つ．待ち行列にあるデータ・パケットは FCFS（first-come/first-serve）の方針で処理するものとする．ここでのサーバによるサービスは，宛先ノードにパケットを送信してから ACK を受け取るまでの処理を指す．サービスにかかる時間は，自ノードのプロセッサの状況と宛先ノードとの通信状況に依存するので，i.i.d.（independently and identically distributed）を仮定する．ここで，単位時間にサービスできる数，つまり，サービス率を μ とする．データの送信には，ある最低の時間がかかるが，以下での解析では分布の裾野に注目するので，最悪のケースを表現するために指数分布を仮定することにする．なお，最下流では，宛先として m 未満の候補しか持てない場合が生じるが，m が小さい範囲を議論するので，その影響は小さい．

一方，各ステップにおけるデータの到着については分布を限定せずに，以下では一般的に，前のデータ到着から t 時間以内に次が到着する確率を $\mathrm{P}[X \leq t] = F_X(t)$ とおき，その到着間隔については同様に i.i.d. を仮定して，到着率を λ とする．なお，λ は配信元のルートが送り出すデータ速度とみることができる．

以上の仮定に基づき，各ステージを $G/M/m$ の待ち行列システム[17] とした直列のモデルとして，伝播の様子を表現する．

5.2 待ち時間の分布

宛先候補の数が増えると，相対的に待ち時間が減少する．$G/M/m$ システムでは，キューの待ち時間が t 以下になる確率は，次のとおりである[17]．

$$F_{W_1}(t) = \mathrm{P}[W_1 \leq t] = 1 - be^{-m\mu(1-\beta)t} \quad (1)$$

ここで，b はデータ・パケットが到着したとき，いずれのサーバも処理中である確率，つまり，到着パケットがサーバの準備を待つことになる確率[⋆1]，β はある条件[⋆2]を満たす特性根であり，到着率 λ のポアソン分布に従うなら $\beta = \lambda/m\mu$，すなわち，サーバ占有率（utilization factor per server）となる．

k ステップ合計の待ち時間を計算するには，まず，式 (1) を Laplace-Stieltjes 変換して，次に k ステップ分（$k > 0$）をその累乗で表して，最後に逆変換して，以下を得る．

$$\begin{aligned} F_{W_k}(t) &= \mathrm{P}[W_k \leq t] \\ &= 1 - \sum_{i=1}^{k} \binom{k}{i} (1-b)^{k-i} b^i \\ &\quad \times Q(i-1; m\mu(1-\beta)t) \end{aligned} \quad (2)$$

ここで，$Q(k;t)$ は累積ポアソン分布[⋆3]である．$k = 0$ の場合は，$F_{W_0}(t) = 1$ である．なお，各ステップにおける待ち時間の分布は互いに独立としたが，実際には，前段のキューで待ったパケットは次段でも待たされることが多いので，待ち時間に関して正の相関を持つ．したがって，実際の分布の裾は式 (2) よりも長くなると考えられる．

⋆1 b は，以下のように計算できる[17]．

$$b = \sum_{k=m}^{\infty} a_k = \frac{a_m}{1-\beta}$$

ここで，a_k は，新しいパケットが到着したとき，k 個のデータがシステムにある（処理中あるいは処理を待っている）確率で，$k = m$ の場合は以下とおり．

$$\frac{1}{a_m} = \frac{1}{1-\beta} + \sum_{j=1}^{m} \binom{m}{j} \frac{1}{(1-\gamma_j)C_j} \frac{m(1-\gamma_j)-j}{m(1-\beta)-j}$$

ここで，

$$\gamma_j = f_X^*(j\mu)$$

$$C_j = \prod_{i=1}^{j} \frac{\gamma_i}{1-\gamma_i}$$

である．

⋆2 到着確率 $F_X(t)$ の Laplace-Stieltjes 変換を f_X^* として，

$\beta = f_X^*(m\mu(1-\beta))$

を満たす．解析的に解けなくても，数値計算により β を求めることができる．

⋆3 ポアソン分布を $P(i;t)$ とすると，以下のとおりである．

$$Q(k;t) = \sum_{i=0}^{k} P(i;t) = \sum_{i=0}^{k} \frac{t^i}{i!} e^{-t}$$

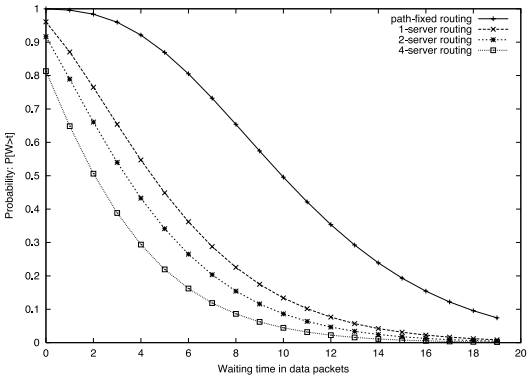

図 6 待ち時間の分布（$n = 8$）
Fig. 6 Waiting time distribution ($n = 8$).

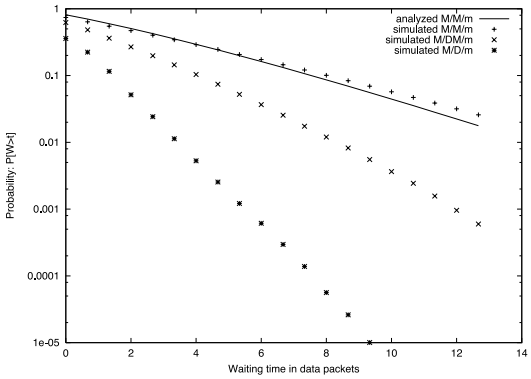

図 7 待ち時間の解析結果とシミュレーションの比較
Fig. 7 Waiting time distribution: analysis and simulation.

さて，あるノードがデータを受信するまでに k ステップの伝播が行われる確率は $\binom{n}{k}/2^n$ である．したがって，最大で n ステップの配信経路において，あるノードがデータを受信するまでに待ち行列に滞在した合計時間は，先の確率で重み付けして以下のように計算することができる．

$$F_W(t) = \mathrm{P}[W \le t] = \frac{1}{2^n} \sum_{k=0}^{n} \binom{n}{k} F_{W_k}(t) \quad (3)$$

図 6 に，最大 $n = 8$ ステップの配信における m-分木の固定配信経路の場合と，$m = 1$，2，4 の宛先候補（サーバ）を持つ再帰的 P2G ルーティングの場合について，キューにおける合計待ち時間の分布を補数で示す．ここでは，ノードに送信性能の上限があることを想定して，サーバ性能の合計（$m\mu$）を一定とするために，サーバ占有率を $\rho = \lambda/m\mu = 2/3$ とした．横軸は，送信データ・パケットの数で表した時間（λt）で，たとえば，$\lambda t = 6$ は配信元から 6 パケットを送信するのにかかる平均時間を表す．固定配信経路の場合，10%のデータ・パケットは $\lambda t = 17.8$ 以上の時間をキューの中で待つことになる．一方，可変配信経路の本手法では，合計処理性能が同じであっても，宛先候補が 1 つ（$m = 1$）の場合で $\lambda t = 11.1$ にまで待ち時間が短縮され，4 つの宛先候補を持つ（$m = 4$）場合にはさらに，$\lambda t = 7.6$ へと大幅に短縮されることが分かる．

図 7 に，解析とシミュレーションの結果を示す．$n = 8$ ステップの伝播，宛先数は $m = 4$，サーバ占有率は $\rho = 2/3$ とした．$M/M/m$ のケースで比較すると，シミュレーションの結果の方がわずかに分布の裾が長くなっていることが分かる．$1/2\mu$ の固定時間と平均 $1/2\mu$ の指数分布の和でサービス時間をシミュレートした結果（$M/DM/m$）を見ると，指数分布によるサービスを仮定した場合よりも，大幅に合計待ち時間が短縮されている．このように，指数分布に従うサービス時間は，期待どおり，最悪のケースを表していることが分かる．

5.3 サービス時間の分布

サービス時間，すなわち中継転送にかかる時間は，それぞれのステップで指数分布に従うとし，また，互いに独立であると仮定しているので，その合計は k ステップ（$1 \le k \le n$）の k-stage Erlangian 分布に従う．ここでは，配信元ルートから 0 番目ステージへの配信時間は含めていない．確率密度関数の Laplace 変換は以下のとおりである．

$$f^*_{S_k}(s) = \left(\frac{\mu}{s+\mu}\right)^k \quad (4)$$

したがって，k ステップ合計のサービス時間が t 以下である確率および確率密度関数は，それぞれ以下のとおりである．

$$F_{S_k}(t) = \mathrm{P}[S_k \le t] = 1 - Q(k-1;\mu t), \quad (5)$$

$$f_{S_k}(t) = \mu P(k-1;\mu t) \quad (6)$$

ここで，$P(k;t)$ および $Q(k;t)$ はそれぞれ，ポアソン分布および累積ポアソン分布である．$k = 0$ の場合は，$F_{S_0}(t) = 1$ である．

ある受信ノードにおいて，データ・パケットが到着するまでにかかったサービス時間の合計が t 以下となる確率は，先と同様に確率 $\binom{n}{k}/2^n$ で重み付けをして，

$$F_S(t) = \mathrm{P}[S \le t] = \frac{1}{2^n} \sum_{k=0}^{n} \binom{n}{k} F_{S_k}(t) \quad (7)$$

で計算できる．また，補数 $F^c_S(t)$ を使って，平均サービス時間は以下のように計算することができる．

$$\overline{S} = \int_0^\infty F_S^c(t)dt = \frac{n}{2\mu} \quad (8)$$

転送 1 段あたりの平均サービス時間が $1/\mu$ なので，m-分木の固定配信経路の最長伝播のケースでは n/μ である．これと比較すると半分の時間となっていること，また，どのノードにおいても，平均的な到達時間が実現されていることが分かる．

5.4 システム時間の分布

データ・パケットが伝播経路に投入されてからノードに到着するまでのシステム時間 T は，待ち時間 W とサービス時間 S の合計で，かつ両者は独立なので，k ステップのシステム時間の分布は，以下のようにして畳み込み積分により計算できる．

$$F_{T_k}(t) = \int_0^\infty F_{W_k}(t-x) f_{S_k}(x)dx \quad (9)$$

ただし，解析的には解けないので，数値計算などの手法を用いて計算する．

受信ノードにおけるシステム時間の分布は，確率 $\binom{n}{k}/2^n$ で重み付けをして，以下のように計算できる．

$$F_T(t) = \mathrm{P}[T \le t] = \frac{1}{2^n}\sum_{k=0}^{n}\binom{n}{k} F_{T_k}(t) \quad (10)$$

図 8 に，最大 $n = 8$ ステップの配信経路で，$m = 1$，2，4 の宛先候補を持つ場合のシステム時間の分布を示す．トラフィック強度（traffic intensity）を $a = \lambda/\mu = 2/3$ とした．横軸は λt で表している．図より，固定配信経路の場合のデータ・パケットの到着が最も遅く，たとえば，90%のデータ・パケットが到着するには，$\lambda t = 23.5$ の時間だけ待つ必要があるのに対して，一方の可変配信経路の場合には大幅に短縮されて，$m = 1$ のとき $\lambda t = 14.6$，$m = 4$ では $\lambda t = 4.9$ の時間内に到着することが分かる．また，この環境設定では，宛先候補が 2 以上になると，ほとんど待ち時間がゼロになっていることも分かる．

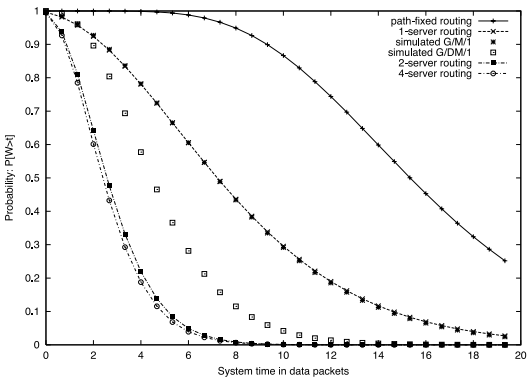

図 8 宛先候補とシステム時間
Fig. 8 System time distribution for $m = 1$, 2, 4 servers.

図には，$m = 1$ の場合についてのシミュレーション結果も同時に示す．指数分布（$G/M/1$）の例は，解析の結果によく一致している．固定時間と指数分布を半々にしたサービス（$G/DM/1$）では，待ち時間が短縮されて，到着時刻が短くなることも分かる．

5.5 エラー回復の解析

データ・パケットの到着を長く待てば，到着率が限りなく向上するように見えるのは，指数分布による転送処理時間を仮定したことによる帰結なので，現実にはあてはまらない．そこで本節では，許容する時間内に到着しなかったデータ・パケットを喪失として扱い，喪失符号により不足分を回復する方法について考察し，小さな誤り率を設計に取り入れる方法について示す．

まず，前方誤り訂正符号（Forward Error Correction: FEC）を導入して，元のデータ・パケットに対して喪失復元のためのパリティ・パケットを追加[4] する．ここで，符号長を B とおき，その中のパリティ・パケットの数を R とおく．この符号ブロックのうち，いずれのパケットが喪失しても，その数が R 以下であるなら元のデータ・パケットを復元することができる．元のデータ・パケットの送出率を λ' とすると，符号化した後のデータ送出率は $\lambda = \lambda' B/(B-R)$ となる．符号化および復号化のためにデータの遅延は少なくとも $2(B-R)$ パケット分が必要である．

符号化すると見かけ上のデータ量が増えるので，到着率 λ でパケットが流れることになる．ここで，それぞれの受信ノードにおいて，復号のために D パケット分のバッファを用意することにする．なお，復号を行うには 1 つの符号ブロック分が必要なので，$D \ge B$ である．このとき，D/λ 時間内に到着しないパケットは喪失したものと見なす．送信元から一定時間間隔でパケットが送信されたとすると，符号語の最初のパケットは時間 $(D-1)/\lambda$ 以内に到着すれば復号に利用できるが，符号語の最後のパケットでは許容時間は $(D-B)/\lambda$ となる．同様に，k 番目（$1 \le k \le B$）のパケットでは許容時間は

$$L_k = (D-k)/\lambda \quad (11)$$

である．したがって，k 番目のパケットが時間内に到着しない確率は以下のように表される．

$$\varepsilon_k = F_T^c(L_k) \quad (12)$$

符号語に含まれるパケットを二分して片方を S とおくと，

$$p_S = \prod_{k\in S}\varepsilon_k \prod_{k\in\overline{S}}(1-\varepsilon_k) \quad (13)$$

表 1 喪失エラー率と設計パラメータ
Table 1 loss error rate and design parameters.

符号長 (B)	冗長性 (R)	許容遅延 (D/D')	宛先候補 (m)	喪失率 (ε_P)
4	1	16/15.0	2	1.49×10^{-6}
4	1	16/15.0	4	3.93×10^{-7}
8	2	16/18.0	2	2.47×10^{-6}
16	4	20/27.0	2	3.43×10^{-6}

は，集合 S のパケットのみが許容時間内に到着しなかったときの確率を表す．したがって，この符号語で回復できない喪失が起こる確率は，符号語あたり，

$$\varepsilon = 1 - \sum_{|S| \le R} p_S \tag{14}$$

で計算される．ここで，$|S|$ は，集合 S の要素の数を表す．データ・パケットあたりでは，

$$\varepsilon_P = 1 - (1-\varepsilon)^{1/(B-R)} \tag{15}$$

となる．この最終的な喪失の確率を設計の目標とした，規模 $n = 8$ および負荷 $\lambda'/\mu' = 2/3$ の配信環境における B，R，D，m の組合せ例を**表 1** に示す．なお，符号化前の送信率 λ' における遅延と符号化のためのバッファによる遅延を合わせると，

$$\begin{aligned} D' &= D(B-R)/B + (B-R) \\ &= (D+B)(B-R)/B \end{aligned} \tag{16}$$

となり，これは実質的な遅延時間を表している．

この表でいえば，$(4, 1)$-符号語を用いて，宛先候補を $m = 2$ として，$D = 16$ のバッファを受信ノードに持たせて復号処理を行うようにすれば，データ・パケットあたりの喪失率を 1.49×10^{-6} に設計できることが分かる．また，宛先候補の多い方が到着時間も短縮されるので，喪失率に関しても良い結果（$m = 2$, 4）になっていることも分かる．25%までの喪失を回復する同じ性能の符号語でも，長ければ（$B = 4$, 8）符号化のための実質的な遅延（D'）が延びるし，また，符号語最後の方のパケットでは許容時間が短く（$D - B$）なるため，かえって喪失率が高くなることが分かる．バースト的な連続した喪失が起こるなら，一般に長い符号語の方が有利であるが，経路が変化する本手法ではエラーは継続せず，したがって，バースト的な喪失も抑えられるので，短い符号語でも効果があると考えられる．なお，符号語を利用せずに，たとえば，$D' = 15$ のバッファ分だけ到着を待つなら，データ・パケットの喪失率は，式 (10) より 1.61×10^{-5} と計算されるが，これは理想的な指数分布の性質に基づいて導出された値であり，現実的にはこのような小さな誤り率の範囲では成り立たない．したがって，喪失符号を利用する方法は，このような小さな誤り率を設計するための現実的な方法である．

5.6 安定性の考察

これまでの解析結果より，遅延や誤り率に関しては，宛先候補の数はたかだか $m = 4$ 程度で十分であることが分かる．もし，データの送信率 λ よりも十分に遅い頻度でノードが離脱するなら，代替ノードの補填（4.4 節）は十分に間に合うので，宛先候補の数をあらかじめ増やす必要はない．しかし，もっと頻繁な宛先ノードの離脱に備えるためには，それ以上の候補を持たせること（$m + s$ の候補）が必要である．

単位時間に λp の割合で各ノードが独立に離脱すると仮定したとき，少なくとも m 個のノードを宛先候補として残すには，p^{s+1} が十分に小さくなるように設計する必要がある．ただし，次のデータ・パケット転送処理までに代わりの宛先候補が補填されることを前提としている．たとえば，10%の離脱率（$p = 0.1$）なら，$s = 5$ 程度の追加ノードがあればよい．このように，宛先候補の数をたかだか 2 倍程度にすれば，配信の性能を維持したままで，十分に安定性も維持できることが分かる．

一方，ルータの故障などが原因となって，ある範囲でまとめて接続が途絶えるような場合には，その方面へのデータ転送はできなくなる．しかし，本手法によれば，それ以外の範囲では引き続き安定してデータ配信を継続することができる．

あるプログラムの開始時刻など，参加ノードが増え続けてゆく場合でも，ネットワーク構成のプロトコル（4 章）上，新規ノードは送信の準備ができるまでは他のノードの宛先候補とはならないので，送信の方向にデータ転送が途絶えるような不完全なネットワークを形成することはない．したがって，安定した配信を続けながら，参加ノードを増やすことができる．

逆に，プログラムの終了時刻など，参加ノードが減少する場合でも，4.3 節で述べたように，離脱の通知とフラッシュ処理に基づく緩やかな離脱処理を行う限り，不完全なネットワークに陥ることなく，安定した配信を継続することができる．

6. 関 連 研 究

アプリケーション層において，対象オブジェクトの位置やそこに至るルートを決定する仕組みとして，Pastry [23] や Tapestry [29] が提案され，その上に pub/sub（publish/subscribe）型でデータを伝え広める SCRIBE [24] や Bayuex [30] が実現された．ここでは，一意の ID に基づいてルーティングが決定する．

また，Castro など[6] は，デカルト超空間に基づいた CAN[22] を利用するフラッディング方式の配信よりも，Pastry や Tepestry など名前のハイパーキューブに基づく木構造の配信方式の方が，配信の性能が高いことを示した．

Yoid[13] は，マルチキャストのための木構造を構築するプロトコルを示し，AMLI[21] は，小さなグループ規模においてノードを相互接続するための最小全域木（minimum spanning tree: MST）を中央集権的に構築するアルゴリズムを提案した．Shi ら[26] は，配信遅延やバンド幅利用率などの観点で解析的に MST の性能を示した．また，Narada[9] では，木構造ではなくメッシュ状の配信経路を構築する．Bullet[18] ではメッシュ状の配信に加えて，複数のソースから同時に受信することによりバンド幅の大きな配信を実現する．

効率の良い配信に加えて，ネットワークの変化に対応するための仕組みもいくつか提案されてきた．NICE[1),2)] では階層的にリーダを固定して配信経路を構成するが，変化に応じてリーダを交代させる仕組みを提案している．中央集権的に状態を監視する Overcast[15] は，ネットワークの変化を中央に効率良く伝える仕組みを提案した．Yang ら[28] は中継ノードの離脱に備えて，あらかじめ予備の候補を用意する方法を提案した．PRM[3] は，データをわずかだけ冗長に別の経路で送るという，経路とデータの両方の冗長性を利用してホスト障害に対応する方法を示した．SplitStream[7] では，データをストライプ状に分割して，それぞれを別の経路に送る，森型の配信を提案した．エラーは一部のストライプだけに影響するとしている．CoopNet[8] は，複数の配信経路に，MDC（multiple description coding）を施して分割したデータを送信することにより，エラー回復を含めた方法を示した．本論文の方法（再帰的 P2G ルーティング）は，経路を複数に固定せず，つねに安定した経路を探しながら配信を行うこと，エラー回復は局所的に送信側主導で行うことに加えて，MDC の一種である前方誤り訂正符号（FEC）を導入して確率的に障害回復を行っていることの 2 点で大きく異なる．

なお，ネットワークの状態や受信ノードの性能によらず，ロバストなデータ配信を実現する仕組みとして，フィードバック・チャネルが不要な，巨大ブロックの喪失符号を利用した方法[5] もある．ここでは，受信ノードはいつでもどのような順番でも，ある必要数のパケットを入手すれば，復号により元のデータを復元できる，ただし，リアルタイム性の必要なデータ配信には向かない．また，ネットワーク上の複雑な変化に対応するための新しい考え方として，Leibnitz ら[19] は，生物の仕組みを模倣して力学系に確率変動を導入する仕組みを提案している，ここでは，1 対 1 通信に複数経路を利用する状況において，ネットワーク変動に合わせて，それぞれの経路へのデータ送信量を動的に変更するために．アトラクタ選択モデル[16] を利用している．生物に学ぶ，新しいアプローチといえる．

7. おわりに

本論文では，オーバレイ・ネットワークを下支えするインフラ（ノード）が不安定であっても，その系において安定したデータ配信を実現するための，新しい端末型マルチキャストの方法（再帰的 P2G ルーティング）を示した．ここでの特徴は，配信の経路を固定せずに，送信先の状況に応じて動的に宛先を変更すること，さらには，安定した状況であっても頻繁に宛先を変更することである．この流動的な経路構成により，性能の低いノードや障害のあるノードは配信経路上の下流に追いやられて，他ノードへの影響が抑えられる．このようにして全体的な安定性を実現する．また，経路の決定は中央集権的に行うのではなくて，各ノードに任せる分散型であるが，階層構造を利用することにより，データのループを生じることはない．エラー回復処理も送信側主導で局所的に行うことにより，配信の規模に対してスケーラブルに行うことができる．さらに，流動的経路構成は，トラフィックの平均化や高速化にも寄与する．

以上の機能をプロトコルの面から示すだけでなく，待ち行列モデルを使ってプロトコルを解析することにより，基本的な性能も定量的に示した．特に，データがノードに到着するまでの「システム時間」を解析し，シミュレーションの結果と合わせて正当性を示した．従来の固定配信経路の方式と比較すると，データ到着までの時間が半分近くにまで短縮される．また，時間内に到着しないデータを喪失として扱い，喪失符号で回復する方法について示した．この考察により，喪失のエラー率を設計に取り入れた配信システムとしてのパラメータ設計が実現できる．安定性については，宛先ノードの数を増やすことで対応できることを示した．

なお，本論文で示した方式に基づいた，1,000 ノード以上の規模の配信システムが，すでに定常的に稼働している．規模をさらに拡大して，また，インターネットなどの異種ノードが混在する環境に拡張していくことにより，配信ルートにおけるノード管理の問題（スケーラビリティ）などに対応してゆく予定である．

謝辞 本研究の方向性に関する明快な議論と有益な

コメントをいただいた松山隆司教授に深く感謝します．また，継続的に議論や実験でおおいに助けてくれた，日本 IBM 東京基礎研究所の中村大賀氏，杉原亮氏，また，実用化に向けて尽力いただいた日本 IBM の関係者の方々に，謹んで感謝の意を表します．

参 考 文 献

1) Banerjee, S., Kommareddy, C., Kar, K., Bhattacharjee, B. and Khuller, S.: Construction of an efficient overlay multicast infrastructure for real-time applications, *IEEE INFOCOM '03*, Vol.2, pp.1521–1531 (2003).
2) Banerjee, S., Bhattacharjee, B. and Kommareddy, C.: Scalable application layer multicast, *ACM SIGCOMM '02*, Vol.32, No.4, pp.205–217 (2002).
3) Banerjee, S., Lee, S., Bhattacharjee, B. and Srinivasan, A.: Resilient multicast using overlays, *ACM SIGMETRICS '03*, Vol.31, No.1, pp.102–113 (2003).
4) Blomer, J., Kalfane, M., Karp, R., Karpinski, M., Luby, M. and Zuckerman, D.: An xor-based erasure-resilient coding scheme, Technical report, International Computer Science Institute, Berkeley, California (1995).
5) Byers, J.W., Luby, M., Mitzenmacher, M. and Rege, A.: A digital fountain approach to reliable distribution of bulk data, *ACM SIGCOMM '98*, Vol.28, No.4, pp.56–67 (1998).
6) Castro, M., Jones, M.B., Kermarrec, A.M., Rowstron, A., Theimer, M., Wang, H. and Wolman, A.: An evaluation of scalable application-level multicast built using peer-to-peer overlays, *IEEE INFOCOM '03*, Vol.2, pp.1510–1520 (2003).
7) Castro, M., Druschel, P., Kermarrec, A.-M., Nandi, A., Rowstron, A. and Singh, A.: SplitStream: High-bandwidth multicast in cooperative environments, *ACM SOSP '03*, pp.298–313 (2003).
8) Chou, P., Padmanabhan, V. and Wang, H.: Resilient peer-to-peer streaming, Technical report, Microsoft Research (2003).
9) Chu, Y.H., Rao, S.G. and Zhang, H.: A case for end system multicast, *ACM SIGMETRICS '00*, pp.1–12 (2000).
10) Deering, S.: Multicast routing in internetworks and extended LANs, *ACM SIGCOMM '98*, pp.55–64 (1988).
11) Deering, S. and Hinden, R.: RFC: 2460: Internet Protocol, version 6 (IPv6) specification (1998).
12) Farley, A.: Minimum Broadcast Network, *Network*, Vol.9, pp.313–332 (1979).
13) Francis, P.: Yoid: Extending the multicast internet architecture, preprint available from http://www.yallcast.com (1999).
14) Hirahara, T., Yamanoue, T., Anzai, H. and Arita, I.: Sending an image to a large number of nodes in short time using TCP, *IEEE ICME '00*, Vol.2, pp.987–990 (2000).
15) Jannotti, J., Gifford, D.K., Johnson, K.L., Kaashoek, F.M. and O'Toole, J.W.: Overcast: Reliable multicasting with an overlay network, *USENIX OSDI '00*, pp.197–212 (2000).
16) Kashiwagi, A., Urabe, I., Kaneko, K. and Yomo, T.: Adaptive response of a gene network to environmental changes by fitness-induced attractor selection, *PLoS ONE*, Vol.1, No.1 (2006).
17) Kobayashi, H.: *Modeling and analysis: An introduction to system performance evaluation methodology*, Addison-Wesley, Reading, Massachusetts (1978).
18) Kosti, D., Rodriguez, A., Albrecht, J. and Vahdat, A.: Bullet: High bandwidth data dissemination using an overlay mesh, *ACM SOSP '03*, pp.282–297 (2003).
19) Leibnitz, K., Wakamiya, N. and Murata, M.: Biologically inspired self-adaptive multi-path routing in overlay networks, *Comm. ACM Archive*, Vol.49, No.3, pp.62–67 (2006).
20) Mathis, M., Semke, J. and Mahdavi, J.: The macroscopic behavior of the TCP congestion avoidance algorithm, *Computer Communications Review*, Vol.27, No.3, pp.67–82 (1997).
21) Pendarakis, D., Shi, S., Verma, D. and Waldvogel, M.: ALMI: An application level multicast infrastructure, *USITS '01*, pp.49–60 (2001).
22) Ratnasamy, S., Francis, P., Handley, M., Karp, R. and Schenker, S.: A scalable content-addressable network, *ACM SIGCOMM '01*, Vol.31, No.4, pp.161–172 (2001).
23) Rowstron, A. and Druschel, P.: Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems, Lecture Notes in Computer Science, Vol.2218, pp.329–350 (2001).
24) Rowstron, A.I.T., Kermarrec, A.-M., Castro, M. and Druschel, P.: SCRIBE: The design of a large-scale event notification infrastructure, *NGC '01*, pp.30–43 (2001).
25) Shastri, A.: Broadcasting in general networks I: Trees, Lecture Notes in Computer Science, Vol.959, pp.482–489 (1995).
26) Shi, S. and Turner, J.: Routing in overlay mul-

ticast networks, *IEEE INFOCOM '02*, Vol.3, pp.1200–1208 (2002).
27) Shimizu, S.: Tree-varying multicast on an overlay network of unreliable end hosts, *IEEE CCNC '04*, pp.158–163 (2004).
28) Yang, M. and Fei, Z.: A proactive approach to reconstructing overlay multicast trees, *IEEE INFOCOM '04*, Vol.4, pp.2743–2753 (2004).
29) Zhao, B.Y., Kubiatowicz, J.D. and Joseph, A.D.: Tapestry: An infrastructure for fault-tolerant wide-area location and routing, Technical Report, UCB/CSD-01-1141, UC Berkeley (2001).
30) Zhuang, S.Q., Zhao, B.Y., Joseph, A.D., Katz, R.H. and Kubiatowicz, J.D.: Bayeux: An architecture for scalable and fault-tolerant wide-area data dissemination, *ACM NOSSDAV '01*, pp.11–20 (2001).

付　　録

A.1 待ち時間の分布

まず，式 (1) を Laplace-Stieltjes 変換すると，

$$f^*_{W_1}(s) = (1-b) + \frac{bm\mu(1-\beta)}{s + m\mu(1-\beta)}$$

この k 乗は，k ステップ分の合計待ち時間の分布を表すので，

$$\begin{aligned} f^*_{W_k}(s) &= \left\{ (1-b) + \frac{bm\mu(1-\beta)}{s + m\mu(1-\beta)} \right\}^k \\ &= (1-b)^k + \sum_{i=1}^{k} \binom{k}{i} (1-b)^{k-i} b^i \\ &\quad \times \left(\frac{m\mu(1-\beta)}{s + m\mu(1-\beta)} \right)^i \end{aligned}$$

これを逆変換して，0 から t の範囲で積分すると，合計の待ち時間が t 以下となる確率を計算できる．

$$\begin{aligned} F_{W_k}(t) &= \mathrm{P}[W_k \le t] \\ &= (1-b)^k + \sum_{i=1}^{k} \binom{k}{i} (1-b)^{k-i} b^i \\ &\quad \times \{1 - Q(i-1; m\mu(1-\beta)t)\} \\ &= 1 - \sum_{i=1}^{k} \binom{k}{i} (1-b)^{k-i} b^i \\ &\quad \times Q(i-1; m\mu(1-\beta)t) \end{aligned}$$

ここで，$Q(k;x)$ は累積ポアソン分布である．

A.2 サービス時間の平均

式 (5) の補数を積分すると，サービス時間の平均を得る．

$$\begin{aligned} \overline{S} &= \int_0^\infty F_S^C(t)dt \\ &= \frac{1}{2^n} \sum_{k=1}^{n} \binom{n}{k} \frac{k}{\mu} \\ &= \frac{n}{2\mu} \end{aligned}$$

ここで，$\int_0^\infty P(i;\mu t)dt = 1/\mu$ より $\int_0^\infty Q(i;\mu t)dt = (i+1)/\mu$ となり，2 行目の変形を得る．また，3 行目は，$(1+x)^n = \sum_{k=0}^{n} \binom{n}{k} x^k$ の両辺を微分して，$n(1+x)^{n-1} = \sum_{k=1}^{n} \binom{n}{k} k x^{k-1}$ として，さらに $x=1$ を代入して変形を得る．

(平成 19 年 6 月 11 日受付)
(平成 19 年 12 月 4 日採録)

清水　周一（正会員）

1962 年生．1985 年京都大学工学部電気工学第 2 学科卒業．1987 年京都大学大学院工学研究科電気工学専攻修士課程修了．同年日本 IBM 入社．東京基礎研究所勤務．2000 年から 2001 年にかけて Princeton 大学客員研究員．3 次元形状 CAD，電子透かし，ネットワーク配信，グリッド・コンピューティング管理等の研究に従事．1994 年度人工知能学会論文賞受賞．