社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2005－DPS－122 (55)
2005－CSEC－ 28 (55)
2005／3／23

多種多様な端末に対する効率のよいビデオ配信方式

山岡 修一 孫 タオ 玉井 森彦 安本 慶一 柴田 直樹† 伊藤 実
奈良先端科学技術大学院大学情報科学研究科 †滋賀大学情報管理学科

本稿では，異なる要求品質をもつ多数のユーザに対するビデオ配信を，オーバーレイネットワーク上で効率よく行う方式 *MTcast (Multiple Transcode based video multicast)* を提案する．MTcast では，オーバーレイネットワーク上の各ノードがビデオストリームを受信，再生しながら同時にトランスコードし，より低品質なストリームを要求するユーザへ中継することで，ユーザ間で異なる様々な要求品質への対応を可能とする．MTcast では，各ユーザは，自分の持つ端末の制約（画面サイズ，利用可能帯域など）に基づいて，ビデオの画像サイズ，フレームレート，ビットレートに対し，それぞれ独立に品質を指定可能である．また，各ビデオセグメントごとに異なった品質を指定することも可能である．全てのユーザは，単一の配送木上に配置され，各々自分の要求品質（もしくは，それに近い品質）でビデオを受信できる．MTcast は，ユーザ数に対するスケーラビリティ，ノードの故障に対する堅牢性を備え，各ユーザは要求に近い品質で，かつ，少ないスタートアップ遅延でビデオの配送をうけることができる．一般的なデスクトップ PC およびノート PC を用いてトランスコードによる負荷を計測したところ，これらの端末が配送木の内部ノードとして十分な性能を達成できることを確認した．また，複数回のトランスコードによる品質の劣化は，十分許容範囲内であることを確認した．

An Efficient Video Delivery Method to Various Terminals

Shuichi Yamaoka Tao Sun Morihiko Tamai Keiichi Yasumoto Naoki Shibata† Minoru Ito
Grad. Sch. of Info. Sci., Nara Institute of Sci. and Tech. †Dept. of Info. Proc. and Man., Shiga Univ.

In this paper, we propose a new video delivery method called *MTcast (Multiple Transcode based video multicast)* which achieves efficient simultaneous video delivery to multiple heterogeneous users by relying on user nodes to transcode and forward video to other user nodes. In MTcast, each user specifies a quality requirement for a video consisting of bitrate, picture size and frame rate based on the user's environmental resource limitation. All users can receive video with specified quality (or near this quality) along a single delivery tree. A different quality requirement can be specified to each video segment. The main characteristics of MTcast are in its scalability, high user satisfaction in received video quality, short startup latency and robustness against node failure. Through experiments, we confirmed that enough performance as an internal node of the delivery tree can be achieved both on a desktop PC and a laptop PC. We also confirmed that deterioration of video quality by repeated transcoding is within permissible range.

1 はじめに

近年のインターネットの爆発的な拡大に伴い，通常の PC に加え PDA や携帯電話端末，車載コンピュータ，情報家電など多種多様な端末が様々な通信回線を経由してインターネット接続されるようになり，処理能力，画面サイズ，利用可能帯域など環境の異なる多数のユーザ端末に対し，ビデオ配信を効率よく行う方法が注目されている．

同一のビデオを異なる品質で複数ユーザに同時配信する方法として，マルチバージョン法 [1, 2]，オンライントランスコード法 [3]，階層化マルチキャスト法 [4] などの様々な方式が提案されてきた．マルチバージョン法は，同じビデオコンテンツに対し，複数のビットレート（品質）のバージョンを複数用意し，ユーザからの要求があった場合に，制約を満たす中でもっとも品質の良いものを選んで配信する．オンライントランスコード法は，オリジナルビデオをサーバあるいは中間ノードで，ユーザが希望する品質のビデオに変換し配信する方法である．階層化マルチキャスト法は，ビデオを階層符号化を用いて基本層および幾つかの拡張層として構成し，ユーザは基本層と任意の数の拡張層を受信しビデオを復号する方法である．各層のデータは独立したマルチキャストストリームとして配信され，各ユーザは制約の許す範囲で複数の拡張層を受信し復号することで制約内で最良の品質のビデオを受信できる．

マルチバージョン法は，制御方式が最も単純で管理が容易であるが，サーバのディスクスペース，ネットワーク帯域の利用効率の面で問題がある．また，マルチバージョン法，階層化マルチキャストでは，バージョンの数，階層の数が十分に大きくない場合には，配信されるビデオの品質がユーザの要求と大幅にずれることもありうる．

本稿では，同じビデオの配信に対し異なる品質要求を持つ多数のユーザに対し，ユーザノードにトランスコードを

行わせ中継させることで，効率良くビデオを同時配信する方法 MTcast (Multiple Transcoding based multicast) を提案する．提案手法では，各ユーザ（ビデオ受信者）は，独自の環境の制約に基づいて決定したビットレート，画面サイズ，フレームレートを要求品質として指定し，ビデオの配信を要求することが可能である．この際，ビデオの時間帯ごとあるいはシーンごとに異なる要求品質を指定することもできる．

提案方式を実現する際には，(1) 高い拡張性：受信ノード数の増加に対応できること，(2) 高いユーザ満足度：各ノードの受信するビデオの品質が，要求した品質に近いこと，(3) 少ない必要資源量：各ノードで実行される，配信システムの制御機構に使用される計算機資源が妥当であること，(4) 短い配送開始遅延：リクエストを送ってから配信が開始されるまでの時間が短いこと，(5) 短い配送遅延：ビデオソースがデータを送信してから末端のユーザノードがデータを受信するまでの時間が短いこと，(5) 適度なトランスコード回数：トランスコード回数の増加による品質低下，配送遅延の増加を防ぐこと，(6) 高い耐故障性：リンク障害やノード障害・離脱が起こっても，配信が停止しないこと，の 6 つの項目を実現することが重要になる．

提案手法では，上記 (1)-(3) を実現するため，コンテンツ送信者を根とするトランスコード木を生成する．トランスコード木は完全 n 分木として構築し，各ノードの計算パワーおよび上り・下り利用可能帯域などの情報を考慮し，高い品質を要求するノードが木の上位に，低い品質を要求するノードが木の低位に，配置される．トランスコード木の各ノード（コンテンツ受信者）は，受信したビデオストリームを実時間トランスコードし，下流のノード（より低い品質を希望する受信者）に転送することで，制約・要求が異なる複数のユーザへのビデオ配信を効率良く実現する．上記 (4)-(6) を実現するため，全受信者を互いに近い品質要求を持つ k 個のノードからなるレイヤと呼ばれるグループ群に分割し，各レイヤをトランスコード木の各ノードとして割り当てることとした．ここで，各レイヤのノードは同一品質のビデオを受け取る．

トランスコード木上のレイヤにのみ注目して取り出した構造をレイヤ木と呼ぶ．レイヤ木の情報は各レイヤから選出した代表ノードに持たせることとした．これにより，新規ユーザは，トランスコード木のどのレイヤが自分の要求品質に最も近いかを容易に探索でき，そのレイヤの親レイヤのいずれかのノードに新規ストリーム配送の要求を迅速に行うことができる．また，トランスコード木の構築の際に，各レイヤにおいて，定められた個数（k から計算される）分新規配送要求に対する計算パワーおよび上り帯域を予備として確保しておくことで，新規ユーザの配送要求およびノード故障に対処している．

新規配送要求，ノード故障への対処により，レイヤの空きが全部埋まってしまった場合，これ以上新規ユーザの要求，ノード故障への処理ができなくなる．提案手法では，空きがなくなった後の要求をバックオーダとして保存しておき，定期的あるいはシーンの切り替わりごとにトランスコード木の再構築を行うことで解決する．バッファリングおよび遅延再生などの方法を組み合わせることで，トランスコード木の再構築の際にもシームレスなビデオ再生が可能になっている．

実際の端末を用いてトランスコードによる負荷を計測したところ，一般的なデスクトップ PC およびノート PC では，トランスコード木の内部ノードとして十分な性能を達成できることを確認した．また，複数回のトランスコードによる品質の劣化は，十分許容範囲内であることを確認した．

以下，2 章では，提案方式によって構築されるビデオ配送木について，その設計方針，構造，構築アルゴリズムを述べる．3 章では，2 章で述べられる配送木において，ノードの加入，離脱，および，再構築処理が発生した際のプロトコルの詳細について述べる．4 章では，提案方式を使用する際のノードの負荷，および，トランスコードによる品質劣化の程度について，実験結果を述べる．5 章では，まとめと今後の課題について述べる．

2 トランスコード木

2.1 定義

ビデオ配信サーバを s，ユーザノードの集合を $U = \{u_1, \ldots, u_N\}$ と表記する．各ユーザノード $u_i \in U$ について，u_i の利用可能上り通信帯域，下り通信帯域が予め決まっていると仮定し，それぞれ，$u_i.upper\_bw$, $u_i.lower\_bw$ と表記する．u_i のビデオの要求品質を $u_i.q$ と表記する．一般に，$u_i.q$ として，ビットレート，画像サイズ，フレームレートの組が指定可能であるが，以降では，簡単のため，ビットレートのみを表すものとする．u_i が品質 q で表されるビデオを同時にトランスコード可能な本数（トランスコード次数とよぶ）を $u_i.n_{transcode}(q)$，同時に転送できる本数を $u_i.n_{link}(q)$ と表記する．ここで，$u_i.n_{tanscode}(q)$ は，u_i の余剰計算パワー P および各品質 q のビデオをトランスコードするのに必要な計算パワー $p(q)$ から，$P/p(q)$ として算出可能である．また，ビデオのビットレートを b とすると，$u_i.n_{link}(q) = u_i.upper\_bw/b$ である．

提案方式では，s を根，U の各要素を節または葉とするオーバレイマルチキャスト木を構築する．以下，本マルチキャスト木をトランスコード木と呼ぶことにする．また，トランスコード木の葉ノード以外のノードを内部ノードとよぶ．

2.2 トランスコード木の構造

トランスコード木の各内部ノードはそれぞれの子ノードへのストリーム配信を担当するが，特定のノードが多くの子ノードを持つことは，負荷が高くなり望ましくない．提案方式では，トランスコード木の各内部ノードが持つ子ノードの数（リンク次数）を，原則として定数値 n とする．

根ノードから送信されたストリームが，実際に葉ノードで再生されるまでの遅延時間およびトランスコード回数を短くするため，トランスコード木が完全 n 分木状（後述の理由により，トランスコード木の根ノードの次数は n ではなく k であるため）になるように構成する．また，トランスコード木の各ノード $u_i \in U$ と u_i の任意の子ノード u_j について，$u_i.q \geq u_j.q$ である．すなわち，トランスコード木の根ノード（動画配信サーバ）から葉ノードへ向かう任意のパス上のノード群の要求品質は降順となるようにトランスコード木を構築する．

トランスコード木の各ノードはユーザの端末であり，任意の時刻において，木から突然離脱する可能性がある．この際，離脱したノード u_l の子ノードへのストリームの中継は，代替ノードによって継続される必要がある．この際，代替ノードは，u_l の再生品質と同程度の再生品質を持つノードの集合から選択されることが望ましい．また，離脱が発生してから代替ノードを探す方法では，u_l の下流ノードでビデオを途切れなく再生することが難しいため，提案方式では，同程度の要求品質を持つ k 個のノードをグループ化しておき，ノードの離脱時には，同じグループ内の別のノードが代替ノードになる方式を採用する．このグループをレイヤと呼ぶ．

トランスコード木のノードを，レイヤに容易に分割できるようにするため，トランスコード木の根ノードの次数のみ k とし（他のノードの次数は n），根ノードの子ノード群に根レイヤを形成させる（図 1）．

提案手法では，全ノードの集合を，レイヤの集合に分割

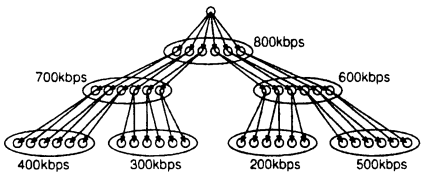


図 1: トランスコード木によるストリーム配信の様子

し，各レイヤ内の全てのノードは，同一品質（レイヤ品質とよぶ）のストリームを親ノードから受信する．また，各レイヤにレイヤ品質の情報を管理するノード（代表ノードとよぶ）を 1 つ選ぶ．トランスコード木上のレイヤの親子関係にのみ注目した木をレイヤ木とよぶ．レイヤ木における各レイヤの子レイヤの数（次数）は，最大で n となる．

$n = 2, k = 6$ のトランスコード木を用いた，ある状態におけるストリーム配信の様子を図 1 に示す．図の円，楕円はそれぞれノード，レイヤを表している．図中のビットレートは，各レイヤのレイヤ品質である．

2.3 トランスコード木の構築

提案方式では，トランスコード木の計算はあるノード u_c で集中的に行う（u_c の決め方は後述）．u には，根ノード s およびその他のノードの集合 $U' \subseteq U$ が予め与えられるとする．

まず，全ノードの集合 $A = \{s\} \cup U'$ を内部ノードの集合 U_I と葉ノード U_L に分類する．なお，s は無条件で U_I に属するものとする．

$$u.n_{transcode}(u.q) \geq 1 + \alpha \quad (1)$$

$$u.n_{link}(u.q) \geq (n+1)(1+\alpha) \quad (2)$$

上記の 2 つの条件式を同時に満たすノード $u \in U'$ を U_I に分類し，満たさないものを U_L に分類する．ここで，式 1 は，u のトランスコード次数が 1 以上かどうか，式 2 は上り帯域が要求ビットレートの $n+1$ 倍以上であるかどうかをあらわす．ここで，α は定数であり，後述の理由により，0.2 程度を想定する．

$|U_I| > |A|/n$ であった場合には，U_I 中の，上り帯域の小さいノードから順に，U_L に移動させ，$|U_I| = |A|/n$ となるようにする．上記の手順により決定された U_I について，$|U_I| < |A|/n$ である場合，U_L に属するノードのうち，上り帯域の大きいノードから順に，上記式 1，2 が成り立つまで，要求品質を減らし，$|U_I| = |A|/n$ となるまで，U_I に移動させる．

次に，全ノードの集合 A をレイヤに分割する．内部ノードの集合 U_I の要素を要求品質 $u.q$ の降順にソートし，上から k 個ずつグループ分けし，内部レイヤとする．この時，グループの最初のノードをそのレイヤの代表ノード とする．レイヤ品質は，レイヤに属するノードの要求品質の平均値とする．葉ノードの集合 U_L も同様に要求ビットレートの降順にソートし，上から k 個ずつグループ分けし，葉レイヤとする．

最後にトランスコード木を構築する．まず，内部レイヤをレイヤ品質の降順でソートし，各レイヤのレイヤ品質が親レイヤのレイヤ品質を超えないように，各内部レイヤを n 分木で接続する．次に，各葉レイヤを，レイヤ品質の降順で最もレイヤ品質の近い内部レイヤの下に接続する（ただし，内部レイヤの子レイヤの数は最大 n とする）．このとき，もし葉レイヤの品質が親レイヤの品質を超える場合は，葉レイヤのレイヤ品質を，親レイヤのレイヤ品質に修正する（修正後のレイヤ品質を実レイヤ品質とよぶ）．

内部レイヤを接続して作成される n 分木（内部レイヤ木とよぶ）の構築方法としては，例えば，レイヤ品質の幅優先探索の順，もしくは，深さ優先探索の順で，各レイヤを内部レイヤ木のノードに割り当てることが考えられる．具体的に，レイヤ数，内部レイヤ数，葉レイヤ数がそれぞれ 15，8，7 のとき，内部レイヤ木を幅優先探索の順および深さ優先探索の順で構築した際の，葉レイヤの実レイヤ品質を図 2 に示す．なお，内部レイヤ，葉レイヤのレイヤ品質はそれぞれ $\{100, 300, \ldots, 1500\}$，$\{200, 400, \ldots, 1400\}$ とした．また，内部レイヤのうち，子レイヤを持たないレイヤ（図 2 のレイヤ品質 100 のレイヤ）は，レイヤ品質ができるだけ小さい内部レイヤの子レイヤとした．

図 2 より，深さ優先探索の順で内部レイヤを構築することで，幅優先探索の場合に比べ，葉レイヤの実レイヤ品質を高くできることが分かる．従って提案方式では，内部レイヤ木をレイヤ品質の深さ優先探索の順で接続することにする．

最後に，内部ノード，葉ノードをそれぞれ要求品質の降順で内部レイヤ，葉レイヤへ割り当てることでトランスコード木を構築する．

2.4 スタートアップ時の手順

ビデオの放送開始時刻を t とする．コンテンツを視聴したいユーザは t 以前に，要求品質および自身のアドレスを含む配信要求をビデオ配信サーバ s に送付する．時刻 t が到来する直前に s は，2.3 節で述べた方法でトランスコード木 T を計算する．また，次回の再構築時にトランスコード木を計算するノード u_c をレイヤの代表ノードの中から決定する．次に，s は，トランスコード木 T および次回計算ノード u_c の情報をレイヤ木に沿って T 内の全てのノードに配布する．すなわち，s は根レイヤの代表ノードに情報を送付し，そのノードは子レイヤの代表ノードに情報を送付するといった操作を葉レイヤの代表ノードに到達するまで繰り返す．また，各レイヤの代表ノードからメンバノードへ必要な情報の配布が行われる．

情報配布の際，各レイヤの代表ノードには，(a) レイヤ木全体の情報（レイヤの親子関係，各レイヤの代表ノードのアドレスとレイヤ品質），(b) 自身のレイヤに属する全ノードのアドレス，配送状況（リンク次数）を持たせる．全てのノードには，(1) 子ノードのアドレス，(2) 所属するレイヤのレイヤ品質と代表ノードのアドレス，(3) 親ノードが所属するレイヤの代表ノードのアドレス，(4) 次回のトランスコード木の計算ノード u_c，を持たせる．

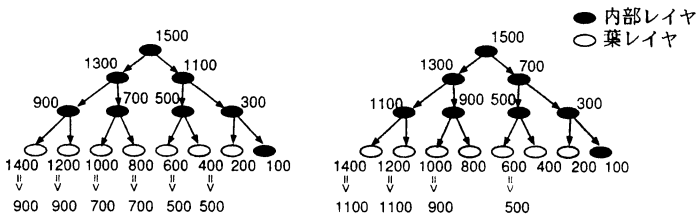
以上により，初期のトランスコード木の情報が全ノードに行き渡り，ビデオの配送の準備が完了する．なお，時刻 t 以降にビデオの配信を希望するノードは，3.1 節で述べる手順で配信を受ける．

2.5 新規ノードの加入と故障への対処

各内部レイヤの各ノードの上り帯域 (レイヤのノードから下位レイヤの各ノードへのデータ転送のための帯域) は，式 2 により，1 本分のストリームを新規に送信できるだけの余裕が持たせてある．新規加入ノードは，この予備帯域を使用して，ストリームを受信する．この際，親ノードのリンク次数が一時的に $n+1$ になることを許している．新規加入ノードに送信するストリームは，既に他の子ノードに送っているストリームと同じストリームを送信するので，新たにトランスコードする必要はない．

また，故障ノードが発生した場合，もしその子ノードのストリームの受信が中断すると，そのノードの下流ノード全てに影響が出る．提案手法では，故障ノードが発生した場合に，子ノードが故障ノードと同じレイヤの他のノードからストリームの受信を継続できるようにするため，前述の予備帯域を使用する．

2.6 節で述べるように，定期的にトランスコード木を再構



(a) 幅優先探索による木の構築　　(b) 深さ優先探索による木の構築

図 2: レイヤ木の構築

築するので，幾つかの新規ノードが加入して予備帯域がなくなっても，再構築の際に，リンク次数 $n+1$ になったノードのリンク次数が n に戻され，予備帯域が新たに確保される．

新規ノードの加入，および，離脱，故障ノード発生時のプロトコルの詳細については，3.1 節，3.2 節でそれぞれ述べる．

2.6 トランスコード木の再構築と切り替え処理

トランスコード木の再構築は，ノード u_c により次の手順で行われる．まず，再構築開始時刻 t_r について，全てのノードが知っていると仮定する．

時刻 t_r が近づくと各ノード u は要求品質 $u.q$ および自身のアドレスを現在のレイヤの代表ノードに送信する．各レイヤの代表ノードは，(子レイヤがあれば) 全ての子レイヤから，また自身のレイヤの全メンバから品質要求を受け取ったら，それを親ノードに送付する．同様に，根レイヤの代表ノードは受け取った情報をノード u_c に送付する．以上により，時刻 t_r 以降に有効となる全てのノードの要求品質の情報が u_c に集まる．

u_c は，2.3 節のアルゴリズムを用いて，トランスコード木を計算し，2.4 節で述べたのと同じ手順で，必要な情報を s から全ノードに配布する．

時刻 t_r になると，全ノードは一旦ストリームの受信を停止し，再構築後のトランスコード木に従って，根ノードから，続きのストリームを順に送信していく．この間，ビデオの再生が途切れないように，各ノードはバッファしておいたデータの再生を行う．

再構築後，次の再構築に備えてバッファを充填する．バッファの充填は，再生速度よりも速い速度でストリームを送信することで行う．バッファの充填を短時間に行おうとすると，ネットワーク帯域とトランスコードのための CPU 資源を消費してしまう．根ノードから葉ノードまでのトランスコード遅延を再構築の間隔で割った値を α とすると，ネットワーク帯域とトランスコードのための CPU 資源は，再生速度と同じスピードで送信・トランスコードする場合に比べ，$(1+\alpha)$ 倍必要になる．

再構築により，トランスコード木の葉ノードに近い位置にいたノードが，根ノードに近い位置に移ったり，逆の移動があり得る．この時に，途切れることなくビデオを再生し続けるため，各ノードがビデオを再生している位置は，いつも全てのノードで同じであるようにする必要がある．したがって，トランスコード木の根ノードに近い位置にいるノードは，葉ノードまでのトランスコード遅延分のストリームをバッファしておく．各ノードの，根ノードからの距離に応じて，バッファしておく量が異なるため，ストリーム受信時に得た統計情報から，再構築の際に，この量を調整する．

表 1: メッセージの詳細

メッセージ	説明
ASK	既存ノードの情報を要求
$ANS(addr)$	既存ノードのアドレスを返す
$REQ(addr, q)$	参加要求
$RESP(addr)$	レイヤの代表ノードのアドレスを返す
$REQ\_V(addr)$	ストリーム配送要求
$RESP\_V$	ストリームの配信
$SEND(addr)$	新規に加入したノードの情報
$BRK(addr, buf)$	親ノードの故障
$COMP$	新規配信ノードの決定
$LEAVE(addr)$	離脱要求
$SEND(addr, q)$	アドレスと要求品質
$UPD(time)$	木の再構築の時刻
$UPD(tree)$	木の再構築に関する情報

3 プロトコルの詳細

本章では，トランスコード木への加入，離脱・故障，再構築時のそれぞれについて，プロトコルの詳細を述べる．

各メッセージの詳細を表 1 に示す．表中の $addr$, $table$, $node$, buf, $time$ をそれぞれノードのアドレス，配信情報テーブル (トランスコード木の構造，各レイヤの品質，各レイヤの代表ノードのアドレス，計算ノードのアドレス)，ノードの情報 (アドレス，要求品質，トランスコード次数)，故障時刻におけるバッファの内容，再構築を行う時刻を表すものとする．

3.1 トランスコード木への加入手続き

加入手続き時の通信手順を図 3 に示す．詳細は次のとおり：(1) ビデオの配信を新たに希望するユーザ u_{new} はロビーサーバと呼ばれる，既にビデオの配信を受けているノード u の情報を持つサーバに u の情報を尋ねる．(2) ロビーサーバから任意に選ばれた u のアドレス $ANS(addr)$ が返される．(3) u に要求品質と自身のアドレスを含めた $REQ(addr, q)$ メッセージを送る．(4) u が代表ノードでなければ，u は受け取ったメッセージを自身の属する代表ノード u_r に転送する．(5) u_r は自身で保持するレイヤ木の情報から，u_{new}・q に近いレイヤ品質を持ち，かつ，新たな配送の予備帯域を持つレイヤを探し，その代表ノード u'_r のアドレスを含むメッセージ $RESP(addr)$ を送信する．(6) u_{new} は u'_r に自身のアドレスを含めたメッセージ REQ_V $(addr)$ を送る．(7) u'_r はレイヤ内の予備帯域を持つノード u' に要求を転送する．

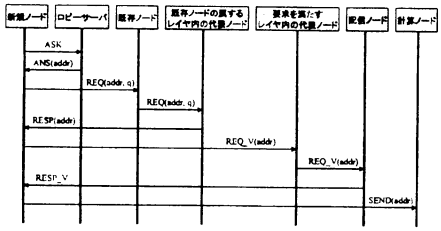


図 3: 新規加入プロトコル

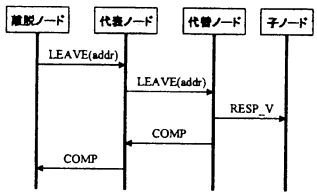


図 4: 離脱時のプロトコル

(8) u' からビデオの配信が開始される.

3.2 離脱・故障ノード発生時の通信手順

ノードに離脱および故障が発生したときの通信手順を, それぞれ図 4, 図 5 に示す. 詳細は次のとおり : (1) 離脱ノード u は, 自分の子ノードのアドレスを含めたメッセージ $LEAVE(addr)$ を親レイヤの代表ノード u_r に送る. (2) u_r はレイヤ内の予備帯域を持つ代替ノード u' にメッセージを転送する. (3) u' は u へのビデオ配信を開始する. (4) 同時に, u' は u_r に配信完了メッセージ $COMP$ を送る. (5) u_r は u にメッセージを転送する. (6) $COMP$ メッセージを受け取った時点で, u は全てのコネクションを切断する.

ノードに故障が発生した場合は次のようになる : (1) ノード u は, 指定したレイヤ品質のデータを一定時間受け取れないとき, 親ノードが故障したものと見なし, 自身のアドレスと受け取れなかった時間分のバッファの内容を含めたメッセージ $BRK(addr, buf)$ を代表ノード u_r に送る. (2) u'_r はレイヤ内の予備帯域を持つ代替ノード u' にメッセージを転送する. (3) u' からビデオ配信が開始される.

3.3 トランスコード木の再構築

トランスコード木の再構築に関する通信手順を図 6 に示す. まず, (1) 計算ノード u_c がメッセージ $UPD(time)$ を全ノード u_a に送信する. (2) 再構築開始時刻が近付くと u_a はメッセージ $SEND(addr, q)$ を u_c に送信する. u_c は 2.3 節のアルゴリズムを用いてトランスコード木を計算し, 2.4 節と同じ手順で代表ノードから子ノードへとメッセージ $UPD(tree)$ を送信する. (3) 再構築により根ノードに近づくノードは再構築後の親レイヤの代表ノードにメッセージ $REQ\_V(addr)$ を送り, ビデオを受信し, バッファリングする.

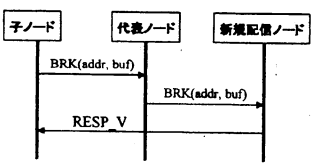


図 5: 故障発生時のプロトコル

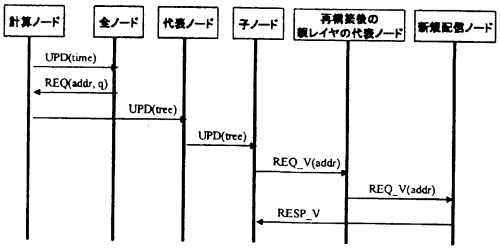


図 6: トランスコード木の再構築プロトコル

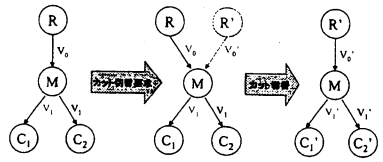


図 7: カット切り替えにおけるバッファ管理

3.3.1 トランスコード木の再構築に伴うバッファリング

木の再構築時にビデオの再生が途切れないようにするため, 各ノードは葉ノードまでのトランスコード遅延分のビデオをバッファしておく必要がある. そこで図 7 のような方法をとる. 中間ノード M が親ノード R からストリーム V_0 を受信し, 子ノード C_1, C_2 へビデオ V_1 をトランスコードして転送していたとする. ある時間においてカット切替の要求を受けた場合, 再構築のタイミング以前にバッファ量を調整するために M は新たに配信を受ける親ノード R' からストリーム V'_0 をカット切替時まで受信してバッファリングする. そして, カットが切り替わると同時に接続を切替, 配信を行う.

4 実験

4.1 トランスコードに伴う負荷

提案手法では, ユーザノード上でトランスコードを行うため, トランスコードによる負荷がビデオの再生に影響しない範囲内であることが求められる. そこで, デスクトップ PC, ノート PC, PDA のそれぞれにおいて, ビデオを再生しながら同時にトランスコードを行う際にどの程度の負荷が生じるかを実験により調べた. 実験では, 各端末上でビデオを再生, トランスコードする際の最大処理速度 (fps) を求め, ビデオの実際の再生速度 (fps) と比較した. 最大処理速度がビデオの実際の再生速度よりも大きければ, トランスコードによる負荷がビデオの再生に影響を及ぼさないといえる. トランスコード次数を 1 から 3 まで変化させ, 最大処理速度の変化を計測した. なお, デコーダとして mpeg2dec 0.4.0b, エンコーダとして ffmpeg 0.4.9-pre1 を用いた.

実験に使用したパラメータと実験結果を表 2 に示す. 表中の各端末の性能はそれぞれ, デスクトップ PC (CPU: Pentium4 2.4GHz, メモリ: 256MB, OS: Linux2.6.10), ノート PC (CPU: Celelon 1GHz, メモリ: 384MB, OS: Linux 2.4.29), PDA (SHARP ZaurusSL-700, CPU: XScale PXA250 400MHz, メモリ: 32MB, OS: Linux 2.4.28) である. また, ビデオのフレームレートは全て 24 fps である.

表 2 より, デスクトップ PC, ノート PC それぞれにおいて, トランスコード次数が 1 であれば, 十分な処理速度を達成できることが分かる. またデスクトップ PC におけ

表 2: 各端末でビデオを再生しながらトランスコードした時の最大処理速度

端末	変換前のビデオ		変換後のビデオ		最大処理速度 (fps)		
					トランスコード次数		
	画面サイズ	ビットレート (kbps)	画面サイズ	ビットレート (kbps)	1	2	3
デスクトップ PC	640x480	3000	480x360	2000	35.66	20.03	14.84
デスクトップ PC	480x360	2000	352x288	1500	61.60	36.40	25.89
ノート PC	352x288	1500	320x240	1000	49.90	30.65	21.84
PDA	320x240	1000	208x176	384	10.12	6.04	4.33

る画像サイズ 480x360, 2000 kbps のトランスコードでは，トランスコード次数が 3 であっても十分な処理速度を達成できている．また PDA に関しては，トランスコード次数が 1 の場合でも最大処理速度が 10.12 であることから，内部ノードとなるのは現実的ではないことが分かる．

4.2 トランスコードによる品質の劣化程度

提案方式では，トランスコード木の根ノードから葉ノードまでの複数回のトランスコードを行うが，ビデオを繰り返しトランスコードすることで画質の劣化を生じるため，劣化の程度が許容範囲内であることが求められる．そこで，あるビデオに対し，ビデオのパラメタ（画像サイズ，フレームレート，ビットレート）の値は変えずにトランスコードを繰り返し行った場合，および，パラメタ値を変化させてトランスコードを行った場合の各々について，品質劣化の程度を，500 フレーム当たりの平均 PSNR 値を測定することで調べた．

図 8 は画面サイズ 640x480，フレームレート 24fps，ビットレート 3000kbps のビデオをパラメタ値を変更せずに繰り返しトランスコードしたときの結果である．トランスコード回数が 3 回までは，若干 PSNR 値が減少しているが，それ以降はほぼ一定の品質を保っていることが分かる．

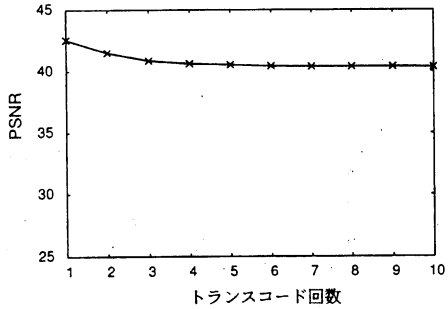


図 8: 同一パラメタへのトランスコードに伴う PSNR 値の変化

次に，表 2 のパラメタ値を使用して，640x480, 24fps, 3000kbps のビデオを，表の 1 行目から 4 行目まで，品質の降順で最大 4 回のトランスコードによる変換を行った場合と，640x480, 24fps, 3000kbps のビデオを，表の各行の品質へ，1 回のトランスコードで変換した場合の PSNR 値の違いを測定した．測定結果を図 9 に示す．図 9 より，複数回のトランスコードにおける PSNR 値は，1 回のトランスコードによる PSNR 値とほぼ同等であることが分かる．

以上の実験より，複数回のトランスコードによる品質劣化への影響は十分許容範囲内であるといえる．

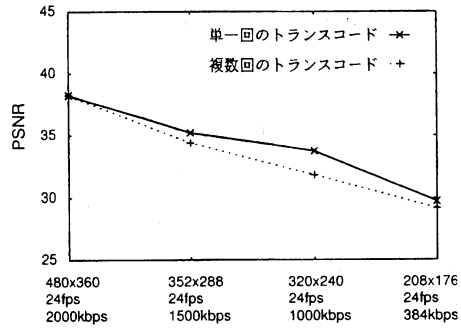


図 9: 異なったパラメタ値へのトランスコードを単一回で行った場合と複数回で行った場合の PSNR 値の差

5 おわりに

本稿では，異なる要求品質を持つ多数のユーザに対し，オーバーレイネットワークを介して効率よくビデオ配信を行う方式 MTcast を提案した．実際の端末を用いてトランスコードによる負荷を計測したところ，デスクトップ PC およびノート PC では，トランスコード木の内部ノードとして十分な性能を達成できることを確認した．また，複数回のトランスコードによる品質劣化の影響は十分許容範囲内であることを確認した．

今後，提案方式を実際の端末上にミドルウェアとして実装し，性能評価を行うことを予定している．

参考文献

[1] G. Conklin, G. Greenbaum, K. Lillevold, and A. Lippman. Video Coding for Streaming Media Delivery on the Internet. IEEE Transactions on Circuits and Systems for Video Technology, 11(3), March 2001

[2] K. Nichols, S. Blake, F. Baker, and D.L. Black, "Definition of the DiRerentiated Services Field (DS Field) in the IPv4 and IPv6 Headers," IETF RFC 2474, Dec. 1998.

[3] S. Jacobs and A. Eleftheriadis. Streaming Video using Dynamic Rate Shaping and TCP Flow Control. Visual Communication and Image Representation Journal, January 1998. (invited paper).

[4] J. Liu, B. Li, and Y.-Q. Zhang, "An End-to-End Adaptation Protocol for Layered Video Multicast Using Optimal Rate Allocation," IEEE Transactions on Multimedia, February 2004.