広域ネットワークにおけるNFSを用いた連続メディアデータアクセス方式

山井成良[†1] 浪平大輔[†2] 安倍広多[†3]
下條真司[†4] 松浦敏雄[†3] 村上孝三[†5]

画像・音声などの連続メディア情報のアクセス方法として，NFSを用いる方式が提案されている．しかし，この方式は広域ネットワークなどの伝送遅延やジッタが無視できない環境では，たとえ十分な帯域があったとしてもスループットが低下する，実時間性が損なわれるなどの問題があった．そこで本論文では，これに対処するために，クライアントと同じLAN内にプロクシを配置する方式を提案する．本方式ではサーバ・プロクシ間の通信をNFSからストリーム型プロトコルに変換することにより上記の問題を解決する．実験の結果，本方式では伝送遅延が大きなネットワークでもスループットが低下せず，また必要なバッファ量も十分小さく，本方式の有効性を確認した．

A Continuous Media Data Access Method Using NFS on Wide Area Network

Nariyoshi Yamai,[†1] Daisuke Namihira,[†2] Kota Abe,[†3]
Shinji Shimojo,[†4] Toshio Matsuura[†3] and Koso Murakami[†5]

NFS-based services have been proposed for transferring continuous media data. However, on a network environment like WAN where delays and/or jitters are outstanding, even if it has enough bandwidth, NFS-based services show low throughput and lose realtimeliness. In this paper, we propose a solution for the above problems by introducing a media-proxy near the clients, that converts NFS protocol into an original stream type protocol between the media-proxy and the NFS-server. The result of our experiments confirms that the proposed method shows high throughput but does not require so much buffer even on networks with large delay.

1. はじめに

近年，計算機の高機能化やネットワークの高速化にともない，動画像や音声などの連続メディア（continuous media）情報をネットワークを介して利用するようなアプリケーションが注目されるようになってきた．これらのメディア情報を表示・再生するには，決められた時間内に次々とメディアデータを供給しなければならず，従来のメディアを扱うアプリケーションよりも厳しい要件が課せられる．したがって，このようなアプリケーションの多くは，限定されたネットワーク環境のもとでのみ利用可能であったり，また，メディアデータの形式や伝送プロトコルなども，それぞれ独自に工夫されたものに限定されていた．そのため，専用のアプリケーションプログラム以外では，メディアデータにアクセスできないという問題があった．

これに対して，連続メディア情報のアクセス方法として，多くのオペレーティングシステム（OS）で利用できるNFS（Network File System）[1),2)]を採用した方式（NFS方式）が提案されている[3)]．この方式では，ローカルマシン上のメディアデータファイルを表示・再生できる任意のアプリケーションプログラムをそのまま用いて，ネットワーク上のメディア情報を利用できる．しかし，NFS方式は，LANのように伝送遅延，ジッタおよび伝送誤りがほとんど起きない環境

†1 岡山大学総合情報処理センター
Computer Center, Okayama University
†2 富士通株式会社
Fujitsu Limited
†3 大阪市立大学学術情報総合センター
Media Center, Osaka City University
†4 大阪大学大型計算機センター
Computation Center, Osaka University
†5 大阪大学大学院工学研究科
Graduate School of Engineering, Osaka University

には適用できるものの，イントラネットや広域ネットワークのように伝送遅延やジッタが比較的大きく伝送誤りが無視できない環境には，たとえ十分な帯域を有する場合にもスループットが低下するため適用できない．したがって，提案されているNFS方式も，LANのような限定された環境でしか実質的には利用できなかった．

広域ネットワークにおけるNFSの性能低下の問題については，今まで多くの議論があり，これを解決するためにAFS（Andrew File System）[4]，WWFS（World Wide File System）[5]，WebNFS[6),7)]など多くの広域分散ファイルシステムが提案されている．このうちAFSはNFSのアクセス性能を改善するためキャッシュや複製（replication）などの技法を用い，同一データへの2回目以降の読み書きの性能を向上させている．しかし，連続メディアデータの再生のようにファイルの先頭から順にデータを読み出す場合には，これらの技法は有効でない．WWFSは主にanonymous FTP（File Transfer Protocol）により公開されているファイルをNFSプロトコルを用いてクライアントに提供することを目的としたファイルシステムで，最初にファイルにアクセスしたときには，いったんファイル全体がローカルファイルシステムに複製されてからクライアントに提供される．このため，特に連続メディアデータファイルのような巨大なファイルにアクセスした場合には，複製完了後のアクセスはローカルファイルシステムと同等に高速化されるものの，初回アクセス時には複製完了までかなりの待ち時間を要する点が問題となる．またWebNFSはパブリックファイルハンドルやmulti-component lookupなどの導入により，従来のNFSプロトコルの持つオーバヘッドを軽減し，広域ネットワークにおける性能の向上を図っている．しかし，WebNFSの基本的な仕組みは従来のNFSと同じRPC（Remote Procedure Call）に基づくものであり，依然として広域ネットワークでは性能上の問題が生じる．

以上のようにこれらの広域分散ファイルシステムはいずれも連続メディアデータの再生には問題が生じる．また，たとえこの問題を解決するような広域分散ファイルシステムが存在したとしても，これを用いることはOSやアプリケーションプログラムに特別な機能を必要としないというNFS方式の利点を損なうことになる．

そこで本論文では，十分な帯域を有するが伝送遅延，ジッタあるいは伝送誤りが無視できないようなネットワーク（以下では，代表的に広域ネットワークと呼ぶ）において，特にNFS方式を対象として，メディア情報を保持するサーバと，それらを表示・再生するクライアント間に一種のプロクシ〔以下，これをメディアプロクシ（media proxy）と呼ぶ〕を導入する方式（メディアプロクシ方式）を提案する．これにより，メディアプロクシが広域ネットワークの及ぼす影響を吸収でき，クライアントのオペレーティングシステムやアプリケーションプログラムを変更せずにメディア情報を伝送することが可能となる．

以下，2章では，ここで提案する方式の実現上の問題点をあげ，3章では，その解決策を示す．4章では本方式の1つの実現方法を明らかにし，5章では試作システムに対する性能評価を行う．

2. 広域ネットワークにおけるNFS方式の課題

NFSは多くのプラットフォームで実装され導入も容易であるため，LAN環境ではファイル共有機構として幅広く用いられている．しかし，NFSを広域ネットワーク上での連続メディアデータ伝送に用いる場合，伝送遅延，ジッタあるいは伝送誤りの存在が問題となる．また，NFSを広域ネットワークを介して運用する場合には，管理上の問題点もいくつか存在することが指摘されている．

そこで本章では，広域ネットワークにおけるNFS方式の問題点について述べる．

2.1 性能上の問題点

2.1.1 ジッタおよび伝送誤りによる実時間性の低下

広域ネットワークでは伝送遅延が一定になることは稀で，ルータ，ブリッジなどのネットワーク機器内の輻輳などのためジッタが生じる．この場合，スループットが十分大きくても，ジッタにより実時間性が損なわれるため，動画像などの連続メディアデータを再生する際には，映像の乱れなどの影響を及ぼす可能性がある．

また，ネットワーク上でパケットロスなどの伝送誤りが生じた場合，他の伝送方式では実時間性を重視して伝送誤りを無視するものが多いのに対して，NFS方式ではファイルシステムとしてメディアデータを提供しているため，伝送誤りを補償する．この場合，**図1**のように再送されたパケットが実際に到着した時刻と伝送誤りを起こしたパケットが本来到着する時刻との差が大きなジッタとして観測され，これが実時間性を大きく損なう原因となる．特に，現在最も普及しているNFS version 2ではトランスポート層として信頼性のないUDPを使用し，再送処理をRPCレベルで

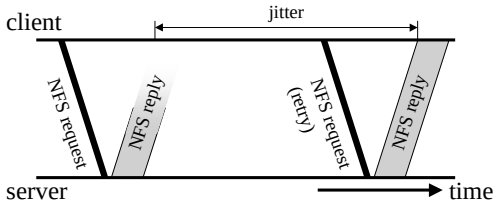

図 1 伝送誤りによるジッタ
Fig. 1 A large jitter caused by transmission error.

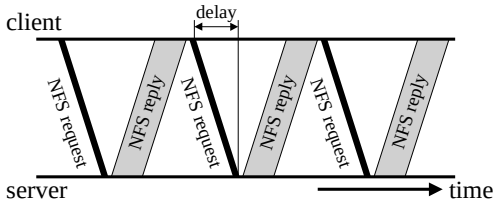

図 2 伝送遅延によるスループットの低下
Fig. 2 Low throughput caused by transmission delay.

行うため，1 つの RPC メッセージが複数のパケットに分割されて送られると☆ 1 つのパケットしか失われていない場合でもメッセージ全体を再送する必要が生じ，パケット単位で再送処理を行う TCP を用いた場合に比べて上記の時間差が大きくなる[8]．

2.1.2 伝送遅延によるスループットの低下

広域ネットワークでは，サーバがクライアントと通信する場合，ルータやブリッジなどのネットワーク機器を経由する．このとき，これらのネットワーク機器では，パケット交換処理のためにある程度（典型的には 1〜3 ms）の伝送遅延が各機器ごとに生ずる．

ところが，NFS は RPC に基づいた stop and wait 型のプロトコルを採用しているため，要求メッセージ間あるいは応答メッセージ間の間隔は伝送遅延時間に大きく依存する（**図 2**）．すなわち，この図において，ネットワークの帯域を B（バイト/秒），伝送遅延時間を d（秒），NFS 要求メッセージの大きさを s（バイト），NFS 応答メッセージの大きさを S（バイト）とすると，この間隔の下限は往復の伝送遅延時間と各メッセージの伝送時間の和である $2d + s/B + S/B$（秒）となり，スループットの上限は $SB/(2\,dB + s + S)$（バイト/秒）と表される．このため，NFS では伝送遅延時間が大きくなると，たとえネットワークの帯域が十分大きくてもスループットが大きくならないという問題がある[9]．多くの実装では，これを補うために 1 ブロック（8 KB）の先読みを行っているが，これによるスループットの改善はたかだか 2 倍であり，伝送遅延時間が大きい広域ネットワークでは不十分である．

☆ NFS での最大メッセージサイズは約 8 KB であるため，データリンク層での MTU（Maximum Transfer Unit）より通常大きく，メッセージの分割は頻繁に発生する．

2.2 管理上の問題点

2.2.1 利用者情報の整合性

NFS では利用者およびグループの識別にそれぞれ UID および GID を用いており，これらを用いてアクセス権の有無を照合している．したがって，これらの利用者情報が異なる異組織間では原則として NFS を運用することはできない．

これに対して，4.4BSD では UID/GID の変換を行う umapfs を NFS と組み合わせてこの問題を解決している[8]．しかし，umapfs ではサーバ管理者が各利用者のクライアント側での UID/GID を知って設定する必要があり，サーバ管理者の負担が大きくなる点が新たな問題となる．

なお，PCNFS[10]ではマウント時に利用者認証を行ってサーバ側の利用者情報を取得するため，この問題は発生しない．

2.2.2 セキュリティ

NFS では本来 LAN 上で用いられることを前提として設計されており，広域ネットワークにおける運用にはセキュリティ上の問題がある．たとえば，NFS ではメッセージの認証が不十分であるため，悪意を持つ第 3 者がメッセージを偽造して不正アクセスを行う危険性がある．また，NFS ではマウント処理のために事前に mount プロトコルを用いてファイルハンドルを取得する必要があるが，その際に用いられるポート番号は portmap サービスによって動的に割り当てられるため，ファイアウォールを導入して不必要なポートへの通信を遮断することが困難となる．

2.2.3 マウント操作

NFS 方式ではアクセスの前にリモートファイルシステムのマウント操作を行う必要があるが，特に UNIX 系 OS ではこのマウント操作には管理者権限が必要であり一般利用者の権限では行えない．この点は，情報コンセントに接続される計算機のように利用者自身が管理者権限を有する場合には問題とならないが，たとえば UNIX 系 OS を搭載した教育用計算機など管理者と一般利用者が異なる環境では，アクセスされる可能性のあるリモートファイルシステムを管理者があらかじめマウントしておく必要があるなど，管理者の負担が大きくなる点が問題となる．

なお，PCNFS では，多くの場合利用者自身がマウント操作を行うことが可能であるため，通常はこの問題は発生しない．

2.2.4 アクセス制御

NFS方式では，サーバ側でマウント要求を受け取った時点でクライアントの計算機名やIPアドレスに基づいてアクセス制御を行い，これを受理するかどうかを決定する．したがって，DHCP環境のようにクライアントのIPアドレスが変化するような環境ではクライアント利用者を特定することができないため，マウント時にクライアント利用者を認証してアクセス制御を行う機能が望まれるが，NFS方式ではこのような機能がない点が問題となる．また，多くの利用者が同時にサーバにアクセスした場合，サービスの品質を保証するために許諾制御（admission control）[11]が必要となるが，NFS方式ではそのような機能が提供されていない点も問題となる．

なお，このうち前者の問題については，PCNFSではマウント時に利用者認証を行うため，問題とならない．

3. プロクシを用いたデータ伝送方式

上記のNFS方式の問題点のうち，性能上の問題点については，広域ネットワーク特有の現象である伝送遅延，ジッタおよび伝送誤りが原因であるため，サーバとクライアントが同一LAN内に存在すれば解消される．そこで，本論文では**図3**に示すようにクライアントと同じLAN内にメディアプロクシと呼ぶ一種のプロクシを配置し，サーバ・メディアプロクシ間の伝送方法を工夫することにより，性能上の問題を解決する方式（メディアプロクシ方式）を提案する．

以下では，本方式において性能上の問題点を解決する方法を示す．なお，管理上の問題点については，本章では議論せず5.4節で述べる．

3.1 バッファの導入

通常，ジッタや伝送誤りによる映像や音声の乱れは，クライアントのアプリケーションプログラム内にあるデータバッファのサイズを大きくすることである程度回避できる．しかし，一般にNFS方式で利用するアプリケーションプログラムは，ファイルがローカルディスク上に存在することを前提として作られているため，データバッファのサイズは小さくジッタや伝送誤りの影響を受けるものが多い．これに対処する方法の1つとしてバッファサイズを十分大きくとるようにアプリケーションプログラムを改造することが考えられるが，この方法では既存のアプリケーションプログラムをそのまま利用できるというNFS方式の利点を損なうことになる．また，NFSの先読みデータ量を増やす方法も考えられるが，この量の変更にはOS自身の修正が必要であったり，あるいは変更そのものは容易であるが十分大きな量を指定できなかったり☆するため，この方法でも十分な対処は困難である．

そこで，メディアプロクシ方式ではクライアントやOSの代わりにメディアプロクシに十分な大きさのバッファを設け，ジッタや伝送誤りの影響を吸収する．また，サーバ・メディアプロクシ間ではフロー制御を行い，バッファ内データの溢れや枯渇を防ぐ．このとき，サーバ・メディアプロクシ間の通信には，フロー制御だけでなく伝送誤り発生時の再送処理やデータの順序保証も求められるため，この通信にはこれらの機能をすべて有するTCPを用いる．これらの手法により，クライアントのオペレーティングシステムやアプリケーションプログラムを改造することなく，ジッタや伝送誤りに対処することが可能となる．

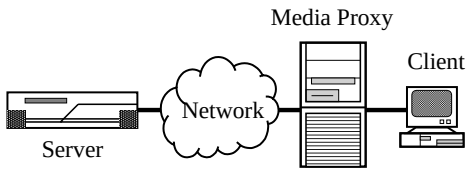

図3 メディアプロクシシステム
Fig. 3 Media-proxy system.

3.2 ストリーム型プロトコルの導入

サーバ・メディアプロクシ間の伝送方式として，従来のNFSプロトコルのようなstop and wait型プロトコルを用いたのでは，2章で述べた理由によりネットワーク帯域が十分ある場合でも必要なスループットが得られない．そこで，本論文では連続メディアデータが多くの場合逐次的にアクセスされる性質に着目し，サーバ・メディアプロクシ間の伝送にストリーム型プロトコルを導入する．すなわち，サーバ・メディアプロクシ間にデータ伝送を指示するPLAYメッセージおよび伝送停止を指示するSTOPメッセージを新たに導入し，メディアプロクシはクライアントから受け取ったREADメッセージをこれらのメッセージに変換するようにする．これにより，サーバは一度メディアプロクシからPLAYメッセージを受け取ると，次にSTOPメッセージを受け取るまで，要求されたデータだけでなくその後続データも連続して伝送することができ，**図4**のように伝送遅延の影響を受けることなく，最大でネットワーク帯域と等しいスループットを得ることが可能となる．

☆ たとえばFreeBSD系UNIXのmount_nfsコマンドでは最大4ブロックまでの先読みしか指定できない．

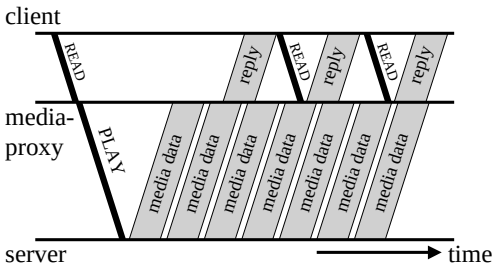

図 4 NFS プロトコルのストリーム型プロトコルへの変換
Fig. 4 Conversion of NFS protocol into stream type protocol.

なお，ストリーム型プロトコルへの変換は READ メッセージだけで十分であり，GETATTR, LOOKUP などの他の NFS メッセージに対してはメディアプロクシはそれを受け取りそのままサーバに中継する．したがって，クライアント・メディアプロクシ間の通信には従来の NFS プロトコルを用いるため，クライアントから見るとメディアプロクシは NFS サーバと等価であり，クライアントのオペレーティングシステムやアプリケーションソフトウェアを改造せずにアクセスすることが可能となる．

4. メディアプロクシ方式の実装

3 章で述べた方針に基づき，我々はメディアプロクシ方式によるシステムを FreeBSD 2.2.2 上で試作した．

試作したシステムでは，クライアント・メディアプロクシのプロトコルとして NFS version 2 を用いることを想定した．これは，同プロトコルだけが利用できる NFS version 3 は利用できないクライアントが存在するのに対して，逆に NFS version 3 を利用できるクライアントは一般に NFS version 2 も利用できるためである．

以下では，試作したシステムの全体構成とメディアプロクシの実装方法について述べる．

4.1 システムの構成

NFS 方式ではサーバ上に NFS 処理を行うプログラム nfsd だけでなく，マウント時にクライアントの認証を行うプログラム mountd が存在する．また，クライアントのオペレーティングシステムの種類によっては，上記の 2 つに加えてユーザ認証を行うために PCNFS[10] 用プログラム pcnfsd が必要となる．

本システムではクライアントがサーバに透過的にアクセスできるように，メディアプロクシプログラム mpd（media-proxy daemon）に上記 3 種類のプログラムで使われる NFS，MOUNT，および PCNFS プロトコルを中継する機能を導入する．また，mpd は

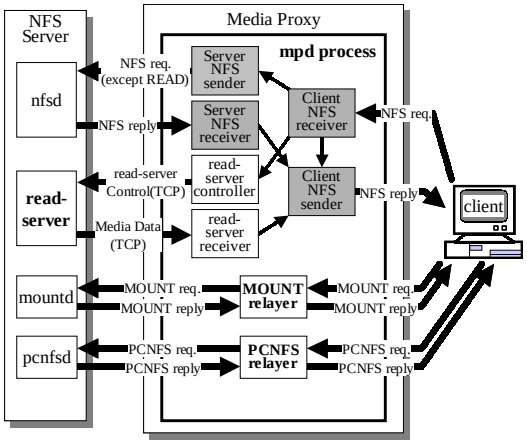

図 5 メディアプロクシの内部構造とメッセージの流れ
Fig. 5 Internal structure and message flow of media-proxy.

クライアントからの NFS メッセージのうち READ メッセージだけを分離し，サーバ上に新たに導入した read-server との間で 3.2 節で述べたプッシュ型プロトコルを用いて連続メディアデータを受信する．READ 以外のメッセージはサーバの nfsd へ単に中継する．

4.2 メディアプロクシの実装

メディアプロクシプログラム mpd では，複数のクライアントからの要求を効率良く処理し，また実時間性をできる限り保証するため，リアルタイムスレッドライブラリ RPTL（realtime portable thread library）[12] を用いている☆．

図 5 にメディアプロクシの内部構造と，メディアプロクシを使用した場合のメッセージの流れを示す．図 5 において，MOUNT relayer および PCNFS relayer は，それぞれ MOUNT および PCNFS プロトコルを中継する．NFS version 2 では UDP が用いられコネクションが存在しないため，NFS メッセージの処理はクライアント数にかかわらず図中の 4 つのスレッド（灰色）で行う．ただし，メディアプロクシと read-server の間は，クライアント数だけ TCP コネクションを確立するため，クライアントごとにスレッドを生成する．このスレッドは MOUNT が成功したときに MOUNT relayer スレッドにより生成される．

クライアントからの NFS 要求は，Client NFS receiver が受け取り，READ 要求以外は Server NFS sender を介してそのまま NFS サーバに中継する．Client NFS receiver が READ 要求を受信すると，

☆ RPTL はユーザレベルで実装されているため，実時間性は必ずしも保証されない．

read-server controller は先読みを行うため read-server に対して PLAY メッセージを送信する．read-server は PLAY メッセージを受信すると，引数で指定されたファイルの送信を開始する．read-server receiver は，受信したデータをバッファに蓄積し，以降のクライアントからの READ 要求に備える．Client NFS sender は以後の READ 要求に対して内部のバッファから応答を返す．なお，read-server と mpd との間のフロー制御は TCP のフロー制御機能により行われる．

複数のクライアントに対して公平にサービスを提供するため，READ 要求の受信と READ 応答の送信は非同期に行われ，client NFS receiver スレッドは READ 要求を受信すると client NFS sender スレッドに READ 応答の送信を依頼する．client NFS sender スレッドは EDF（Earliest Deadline First）スケジューリング方式[13]に基づいて前回の応答時刻から見て最も差し迫っている要求から順に応答を返す処理を行う．

ここで，READ 要求にはファイルハンドルだけが含まれており，ファイル名は含まれていない点が問題となる．すなわち，一般にファイルハンドルからファイル名を得ることは困難であるため，PLAY メッセージで，送信するファイルをファイルハンドルで指定しても，read-server では送信すべきファイルを特定することができない．この問題に対し，mpd では次のようにして対処した．NFS クライアントは READ 要求を行う前に，ファイルハンドルを取得するために，READDIR および LOOKUP 要求を発行する．これらの要求にはディレクトリ名やファイル名の情報が含まれており，応答にはファイルハンドルが含まれている．mpd は中継時にこれを盗み見て，内部にファイル名とファイルハンドルの対応表を作成する．これによって PLAY メッセージでファイル名を渡すことができ，read-server は送信するファイルを特定することができる．

5. 試作システムの性能評価と考察

本論文で提案するメディアプロクシ方式の有効性を検証するため，4 章で述べた試作システムの性能評価を行った．以下では性能評価の方法とその結果について述べる．

5.1 広域ネットワークにおける伝送速度の評価

まず，伝送遅延がスループットに与える影響を調査するために，広域ネットワーク上で試作システムを用いて実験を行った．本実験におけるシステム構成を**図 6** に示す．本実験は平成 9 年 6 月に大阪市で開催された Network Kansai '97 の展示機器を用いて行われた．サーバ・ハブ間のネットワークには Cyber Kansai Project☆の実験用 ATM ネットワーク（155 Mbps）を用いており，途中で ATM 交換機やルータをいくつか経由することにより伝送遅延量を変更することが可能である．また，ハブ（100 Mbps イーサネットハブ）には 2 台のクライアントと 1 台のプロクシを接続した．クライアント 2 台のうち，1 台は NFS 方式でアクセスするように設定し，もう 1 台はメディアプロクシ方式でアクセスされるように設定した．クライアントは 2 台とも CPU が Pentium（75 MHz）で OS として Windows 95 が動作しており，PCNFS クライアントソフトウェアと MPEG2 デコーダボードが装備されている．また，再生アプリケーションは Windows 95 に標準で添付されている Media Player を用いた．メディアプロクシは CPU が Pentium Pro（200 MHz）で OS は FreeBSD 2.2.1 である．

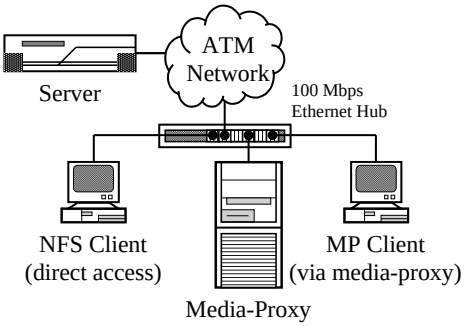

図 6 実験環境 1
Fig. 6 Experiment environment #1.

以上のネットワーク環境においてサーバ・メディアプロクシ間の経路を幾通りかに設定して伝送遅延量を変化させ，各設定ごとに 2 台のクライアント上でそれぞれ 6 Mbps と 3 Mbps で符号化された MPEG2 データを再生した場合のサーバ・メディアプロクシ間の伝送速度を測定した．また，同時に再生された動画像や音声に途切れなどの悪影響が現れていないかを実験者が観測した．

実験の結果得られた RTT（Round Trip Time）とスループットの関係を**図 7** に示す．

この図より，NFS 方式では 6 Mbps，3 Mbps の MPEG2 データの再生において，伝送遅延がそれぞれ 3 ms 以上，18 ms 以上になるとスループットが低下するのに対し，メディアプロクシ方式ではそれぞれ 18 ms 以下，29 ms 以下の範囲では規定のスループット

☆ http://www.ckp.or.jp 参照.

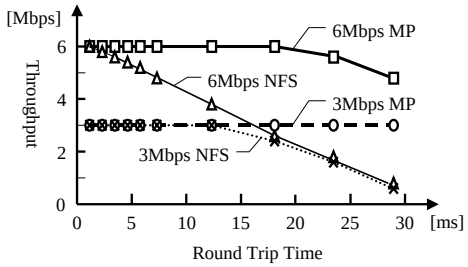

図 7 RTT とスループットとの関係
Fig. 7 Throughput for various RTTs.

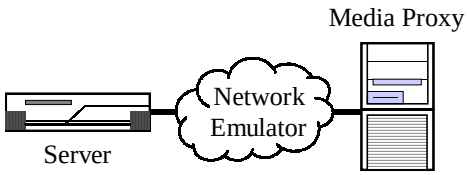

図 8 実験環境 2
Fig. 8 Experiment environment #2.

表 1 岡山大学・大阪大学間のネットワーク特性
Table 1 Network characteristics between Okayama Univ. and Osaka Univ.

平均 RTT	21.1	(sec)
RTT の標準偏差	3.6	(sec)
パケットロス率	0.057	(%)

が得られていることが分かる．また，実験者による観測でも，NFS 方式では伝送遅延がそれぞれ 3 ms 以上，18 ms 以上になると再生された画像や音声に途切れが生じたのに対し，メディアプロクシ方式では 6 Mbps では 18 ms 以下，3 Mbps では測定したすべての設定において途切れが確認されなかった．

以上の結果から，伝送遅延が大きいネットワークにおいてメディアプロクシ方式は NFS 方式の欠点を補い，有効であるといえる．

なお，伝送遅延が 24 ms 以上のときには，メディアプロクシ方式でも 6 Mbps の MPEG2 データの再生においてスループットの低下が見られるが，本実験では TCP のウィンドウサイズがたかだか 16 KB 程度であったことが確認されている☆ことから，この原因はコネクションの遅延帯域積と比較して TCP ウィンドウサイズが小さかったためと考えられる．これは，ウィンドウサイズが 16 KB の場合，RTT が 24 ms, 29 ms のときの理論的なスループットの上限（ウィンドウサイズを RTT で割った値）はそれぞれ 5.3 Mbps, 4.4 Mbps となり図 7 の結果とよく一致することからも裏付けられる．

5.2 必要なバッファ量の評価

次に，ジッタや伝送誤りの大きさとこれらの影響を吸収するために必要なバッファ量との関係を明らかにするために実験を行った．

実験環境を**図 8** に示す．この図において，サーバ・メディアプロクシ間にはネットワークエミュレータ nistnet [14]が設置され，伝送遅延時間，ジッタの大きさおよびパケットロス率を設定できるようになっている．また，クライアントの代わりにメディアプロクシ自身が 40 ms ごとにバッファ内から 8 KB のデータを読み出し☆☆，バッファの枯渇を検出できるようにしている．

実験に先立ち，対象となるネットワークの特性を決定するために，岡山大学・大阪大学間のネットワーク特性を測定した．その結果を**表 1** に示す．なお，表 1 における平均 RTT と RTT の標準偏差は，ping コマンドを用いて 1000 個の 64 バイトパケットを送出して測定したものであり，また，パケットロス率については DBS [15]を用いて約 1.5 Mbps の帯域の MPEG データを疑似的に 5 分間送出して測定したものである．

この結果に基づき，伝送遅延時間（RTT の半分）の平均値，伝送遅延時間の標準偏差，パケットロス率の標準値をそれぞれ 10 ms，2.5 ms，0.05%とし，それぞれの特性を標準値を中心として 3 通りに変化させ，すべての組合せについて実験を行った．具体的には，平均伝送遅延を 0 ms，10 ms，20 ms の 3 通り，ジッタの大きさ（伝送遅延の標準偏差）を 0 ms，平均伝送遅延の 4 分の 1，平均伝送遅延の 2 分の 1 の 3 通り，パケットロス率を 0%，0.05%，0.5%の 3 通りに変化させた．各組合せにおいては，手持ちのビデオファイルの標準的な再生時間が約 5 分間であったことから，1.5 Mbps の帯域で 5 分間分に相当する約 56 MB のファイルを用意し，これを 10 回再生して各再生において使用したバッファ量を測定した．各組合せにおいて必要バッファ量（平均値 + 1.64 × 標準偏差）☆☆☆を算出した結果を**図 9**，**図 10**，および**図 11** に示す．これらの図において，縦軸は NFS のブロックサイズ（8 KB）単位で表したバッファの大きさである．なお，図 11 において平均伝送遅延が 20 ms，伝送遅延の標準偏差が平均伝送遅延の 2 分の 1 の場合の値は，必要な帯域が得られなかったため表示していない．

これらの図より，以下のような傾向が読みとれる．

☆ ハブにおける輻輳が原因であると思われる．
☆☆ 1.5 Mbps の帯域に相当する．
☆☆☆ 使用バッファ量の分布が正規分布に従うと仮定した場合に 95%以上の確率で枯渇が生じないバッファ量．

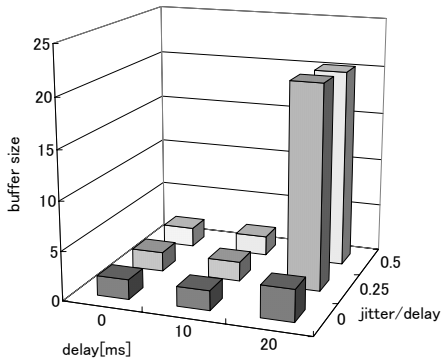

図 9 パケットロス率が 0%の場合の必要バッファ量
Fig. 9 Required buffer size (packet loss rate = 0%).

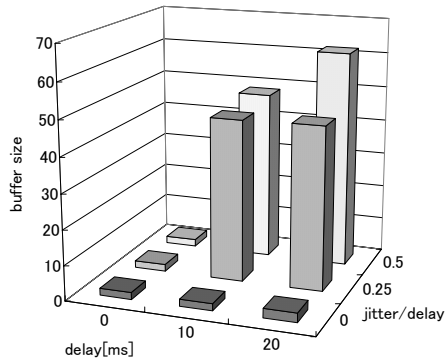

図 10 パケットロス率が 0.05%の場合の必要バッファ量
Fig. 10 Required buffer size (packet loss rate = 0.05%).

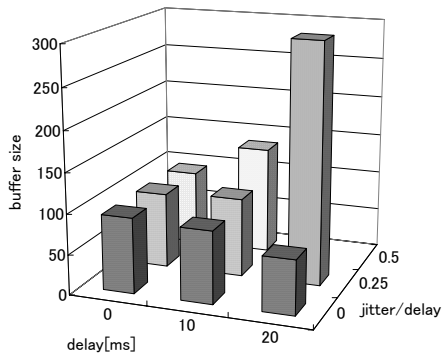

図 11 パケットロス率が 0.5%の場合の必要バッファ量
Fig. 11 Required buffer size (packet loss rate = 0.5%).

ネットワークの状態がある程度良い場合には，必要なバッファ量は非常に少ない（2ブロック程度）が，ネットワークの状態が悪化すると数十ブロック程度にまで必要なバッファ量が急激に増加する．その後，ネットワークの状態が悪化するにつれて必要なバッファ量は100ブロック前後にまで漸増していくが，さらに悪化すると最終的には帯域不足となり，その直前の状態では非常に大きいバッファ量（本実験では300ブロック程度）が必要となる．

これらの現象は以下のように説明できる．ネットワークの状態がある程度良い場合には，TCPの再送処理がほとんど不要であるか，必要であっても短時間のうちに再送処理が行われるため，小さいバッファ量（2ブロック程度）で十分である．ところが，ネットワークの状態が悪化すると，たとえばあるパケットが失われ，さらに再送されたパケットも失われるなどの理由により再送処理に比較的長時間を要するようになり，数十〜100ブロック程度の比較的大きいバッファが必要になる．ネットワークの状態がさらに悪化すると，帯域に余裕がなくなるため，一度再送処理が発生するとバッファ中のデータを大量に消費したまま回復しない状態で次の再送処理が発生し，結果として非常に大きいバッファ量（300ブロック程度）が必要となる．

以上の結果および考察より，実験した環境では標準的な状態からネットワークの状態が多少悪化しても必要なバッファ量は大きく変化せず，クライアント1台につき100ブロック（800 KB）程度のバッファを用意すれば十分であり，現在の計算機に搭載されている主記憶容量と比較すると十分小さいといえる．

なお，パケットロス率が0.5%でジッタがない場合，平均伝送遅延の増加に従って必要バッファサイズが減少する傾向が見られるが，これはたとえばパケットロスが短時間に複数回発生した場合，平均伝送遅延が小さいと個別に再送処理が行われTCPのスループットが大きく抑制されるが，平均伝送遅延が大きいとまとめて再送処理が行われ，スループットがそれほど大きく抑制されないことが原因であると考えられる．

5.3 接続可能なクライアント数に関する考察

試作システムでは，1台のメディアプロクシに対して複数のクライアントを接続する構成であるため，接続可能なクライアントの台数が問題となりうる．そこで，以下ではこの問題について考察する．

本システムの構成では，クライアントの台数が増加するに従い，サーバ・メディアプロクシ間の通信量が増えメディアプロクシの負荷が大きくなる．しかし，通信量の上限はサーバの能力やサーバ・メディアプロクシ間のネットワーク帯域によって定まるため，たとえメディアプロクシの能力が十分高い場合でも接続可能なクライアント数は通信量の上限により制限されることになる．また，メディアプロクシの処理能力についても上記の通信量を処理する能力を有すれば十分であると思われる．ただし，5.2節で示したように，本システムではネットワーク帯域に余裕がない状態では

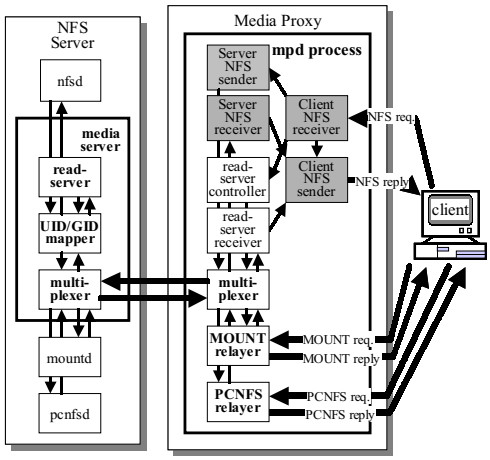

図 12 PNFS の機能を組み込んだメディアプロクシの構成
Fig. 12 Revised structure of media-proxy with PNFS functions.

必要バッファサイズが大きく増加するため，実用上接続可能なクライアント数は上記の制限より多少減少すると思われる．なお，このような制限は他の伝送方式でも同様に存在し，メディアプロクシ方式固有の問題ではないと思われる．

5.4 管理上の問題への対処

試作システムは，主にメディアプロクシ方式の性能面での有効性を検証するための機能が実装されており，2.2 節で述べた管理上の問題に関しては特に考慮されていない．そこで本節では管理上の問題に対して対処する方法について述べる．

我々は，2.2 節で述べた問題に対処するため，別途に NFS に基づいた広域分散ファイルシステム PNFS [16] を開発している．PNFS では UID/GID の自動取得・相互変換機能やメッセージの認証・暗号化機能を実現しており，メディアプロクシ方式に PNFS の機能を組み込むことにより管理上の問題に対処することが可能であると思われる．

PNFS の機能を組み込んだ場合のメディアプロクシシステムの構成を**図 12** に示す．この構成ではサーバ側にも NFS，PCNFS，MOUNT の各プロトコルを中継するプロクシ（メディアサーバ）を配置し，メディアサーバ・メディアプロクシ間は 1 つのコネクションだけを用いてこれらのプロトコルを多重化して通信する．multiplexer はこの多重化を担当するが，さらに必要に応じてメッセージの認証や暗号化も行う．マウント時には MOUNT プロトコルで用いられる引数だけでなく，利用者名やパスワードなどの認証情報もメディアプロクシからメディアサーバに送られ，メディアサーバでは利用者認証に成功すると認証情報に基づき UID/GID 変換機能を設定する．マウント後の NFS 要求メッセージは UID/GID 変換機能によりサーバ側の UID/GID に変換された後に nfsd などのプログラムに渡される．

以上の動作により，NFS 方式を広域ネットワークを介して運用する際の利用者情報の整合性の問題を解決することができ，またセキュリティの強化も実現できると思われる．さらにファイアウォールを併用した運用についても，メディアサーバ・メディアプロクシ間の通信を特定の固定ポート番号を用いて行うことが可能となるため，何ら問題が生じないものと思われる．

また，マウント操作の権限や利用者認証によるアクセス制御の問題については，PNFS ではサーバ側の利用者名とパスワードを入力すれば，一般利用者でも利用者認証後にマウント操作を行える機能を持つマウントプログラムを提供しており，メディアプロクシ方式においても同等の機能を導入することにより解決できると思われる．

残された問題として，多くの利用者が同時にサーバにアクセスした場合における許諾制御があるが，メディアサーバではサーバ・メディアプロクシ間の現在の通信量やネットワーク帯域を推定できるため，これらの推定結果に基づいて read-server への新たな再生要求を受理あるいは拒否する機構を導入すればこの機能も実現できると思われる．

6. むすび

本論文では，帯域は十分にあるものの伝送遅延やジッタが無視できないようなネットワークにおいて NFS 方式の問題点を解決する連続メディアデータ伝送方式として，クライアントが接続されている LAN 内にメディアプロクシを配置するメディアプロクシ方式を提案した．また，実際に本方式に基づいたシステムを実装・構築して性能評価を行い，性能の面では本方式が実用上十分な性能が得られること，ならびに想定しているネットワーク環境では必要なバッファ量も十分小さく実現可能性の面でも問題のないことを確認した．また，管理の面についても従来の NFS 方式の問題点に対する解決策を示した．これにより，帯域が十分にある広域ネットワークを介した環境においても LAN 環境と同様に，NFS 方式の利点を活かした連続メディアデータアクセスが可能となる．

しかし，本方式は連続メディアデータをそのまま伝送しているため，サーバ・メディアプロクシ間の帯域がつねに十分大きいネットワークでなければ利用でき

ない．そこで，今後の課題としては，利用可能な帯域に応じてQoS（quality of service）を動的に制御する機構をファイルシステムとして提供することがあげられる．

謝辞 本研究にあたって種々の面でご援助いただいた松下電器産業（株）マルチメディア開発センター岡秀幸氏をはじめ，同センター関係者の皆様に厚く感謝する．また，試作システムの評価実験にご協力いただいたサイバー関西プロジェクトの関係者の方々に深謝する．

参 考 文 献

1) Sun Microsystems, Inc.: NFS: network file system protocol specification, Technical Report RFC1094, IETF (1989).
2) Callaghan, B., Pawlowski, B. and Staubach, P.: NFS version 3 protocol specification, Technical Report RFC1813, IETF (1995).
3) 大村 猛，廣田照人，中西正典，岡 秀幸：ビデオサーバソフトウェア—Video Network Server—その設計と実装評価，電子情報通信学会論文誌（D-II），Vol.J79-D-II, No.4, pp.626–633 (1996).
4) Howard, J.H.: An Overview of the Andrew File System, *Proc. USENIX Conference*, pp.23–26 (1988).
5) Kadobayashi, Y., Yamaguchi, S. and Miyahara, H.: WWFS: An Evolutionary Framework for File Sharing in the Internet, *Proc. INET'93*, San Francisco, U.S., pp.DGB-1–DGB-8 (1993).
6) Callaghan, B.: WebNFS Client Specification, Technical Report RFC2054, IETF (1996).
7) Callaghan, B.: WebNFS Server Specification, Technical Report RFC2055, IETF (1996).
8) McKusick, M.K., Bostic, K., Karels, M.J. and Quarterman, J.S.: *The design and implementation of the 4.4BSD operating system*, Addison-Wesley, Reading, MA. (1996).
9) 中島康之，氏原清乃，米山暁夫：CATV回線上のLANを用いたVOD実験，1996年電子情報通信学会ソサイエティ大会講演論文集，B-655 (1996).
10) X/Open Company, Ltd.: *Protocols for X/Open internetworking: (PC) NFS*, United Kingdom (1991).
11) Tanenbaum, A.S.: *Computer networks*, 3rd edition, Prentice-Hall, NJ (1996).
12) 安倍広多，松浦敏雄，安本慶一，東野輝夫，谷口健一：UNIX上で周期スレッドを実現するユーザレベルスレッドライブラリの実現法，電子情報通信学会技術研究報告，CPSY-97-24, pp.49–54 (1997).
13) Ramamritham, K. and Stankovic, J.A.: Scheduling algorithms and operating systems support for real-time systems, *Proc. IEEE*, Vol.82, No.1, pp.55–67 (1994).
14) Parker, S. and Schmechel, C.: Some Testing Tools for TCP Implementors, Technical Report RFC2398, IETF (1998).
15) 村山公保，門林雄基，山口 英：TCP性能評価システムDBSの構築，コンピュータソフトウェア，Vol.15, No.2, pp.24–37 (1998).
16) Nakayoshi, K., Yamai, N., Matsuura, T., Abe, K. and Murakami, K.: A secure private network file system with minimal system administration, *Proc. 1997 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing* (*PACRIM'97*), Vol.1, Victoria, Canada, pp.251–255, IEEE (1997).

(平成12年 5 月8日受付)
(平成12年12月1日採録)

山井 成良（正会員）

昭和59年大阪大学工学部電子工学科卒業．昭和61年同大学大学院博士前期課程修了．昭和63年同大学大学院基礎工学研究科（物理系専攻情報工学分野）博士後期課程退学．同年奈良工業高等専門学校情報工学科助手．同講師，大阪大学情報処理教育センター助手，同大学大型計算機センター講師を経て，現在岡山大学総合情報処理センター助教授．分散システム，マルチメディアシステム，マルチメディアネットワークの研究に従事．IEEE，電子情報通信学会各会員．博士（工学）．

浪平 大輔（正会員）

平成8年大阪大学工学部情報システム工学科卒業．平成10年同大学大学院工学研究科前期課程修了．同年富士通（株）入社．高速/高品質ネットワークに関する製品開発に従事．

安倍 広多（正会員）

平成4年大阪大学基礎工学部情報工学科卒業．平成6年同大学大学院博士前期課程修了．同年NTT入社．平成8年大阪市立大学学術情報総合センター助手．平成12年同講師．博士（工学）．マルチスレッド機構の実装，オペレーティングシステムの設計等に興味を持つ．電子情報通信学会会員．

下條　真司（正会員）

昭和 61 年大阪大学大学院基礎工学研究科博士後期課程修了．同年大阪大学基礎工学部情報工学科助手，平成元年同大学大型計算機センター講師，平成 3 年同助教授，平成 11 年同教授現在に至る．分散処理システムの設計開発，マルチメディアシステム，オブジェクト指向データベース等の研究に従事．工学博士．電子情報通信学会，IEEE，ACM 各会員．

松浦　敏雄（正会員）

昭和 50 年大阪大学基礎工学部情報工学科卒業．昭和 54 年同大学大学院基礎工学研究科（情報工学専攻）博士後期課程退学後，同年大阪大学基礎工学部情報工学科助手．平成 4 年同大学情報処理教育センター助教授．平成 7 年大阪市立大学生活科学部教授．平成 8 年同大学学術情報総合センター教授現在に至る．工学博士．ソフトウェア開発環境，ユーザインタフェース，マルチメディア，情報教育等に興味を持つ．ACM，IEEE，電子情報通信学会各会員．

村上　孝三（正会員）

昭和 46 年大阪大学工学部電子工学科卒業．昭和 48 年同大学大学院修士課程修了．同年富士通（株）入社．富士通研究所通信研究部門，同所マルチメディアシステム研究所を経て，平成 7 年大阪大学大型計算機センター教授．平成 10 年同大学大学院工学研究科情報システム工学専攻教授．マルチメディア情報通信システム，インテリジェントネットワーク，フォトニックネットワークの研究に従事．工学博士．IEEE シニア会員，電子情報通信学会会員．