システムソフトウェアと
オペレーティング・システム 78－17
（1998．5．8）

ネットワークエンドホストのための QoS 調停 システム

松井 康範† 木原 誠司† 盛合 敏† 徳田 英幸‡
NTT 情報通信研究所†
慶應義塾大学環境情報学部‡

次世代の分散マルチメディア環境では、一つのホストが複数の連続メディアデータストリームを同時に扱うようになると考えられる。このように急増するアプリケーションの帯域要求に対して、限定された資源を有効に利用する枠組が求められている。帯域を共有しながら有効に利用するためには、動的に QoS を適応させることのできる、柔軟なネットワーク制御を行なわなければならない。
本論文では、VBR 連続メディアの転送に適したネットワークサブシステムのフレームワーク VoR を提案する。VoR では QoS を変換・調停するミドルウェアとともにネットワークドライバ内のスケジューラが動的な帯域割り当てを行なう。また、アプリケーションプログラムがネットワークサブシステムの状態情報を得たり、メディアスケーリングなどに反映させることができるようなメカニズムも提供している。
提案したシステムを RT-Mach および ATM ネットワークカードを利用して実装し、実験評価を行なった。

QoS Mediation System for Network End Hosts

Yasunori Matsui†, Seiji Kihara†, Satoshi Moriai† and Hideyuki Tokuda‡
NTT Infomation and Communication Systems Labs†
Keio University, Faculty of Environmental Information‡

Forthcoming distributed multimedia systems are required to handle multiple continuous media streams simultaneously. Even if the network speed increases, the last-one-mile problem is expected to remain by the expanding demand for bandwidth. In order to share and effectively utilize limited resource, dynamic QoS mediation is necessary. Dynamic adaptation and flexibility is the key in the system design.
In this paper, we propose VoR system, a network subsystem framework for VBR continuous media transmission. With the help by QoS coordinator middleware, the scheduler in the network driver multiplexes continuous media streams with QoS specification. VoR also provides a mechanism for application programs to inspect and interact with the underlying network subsystem.
We have implemented the proposed subsystem on RT-Mach OS and ATM. Some results of its performance and characteristics are presented.

1 はじめに

次世代のインターネットは、高品質な連続メディア通信をサポートすることが強く期待されている。Gigabit-Ethernet, ATM などの高速ネットワークの出現、Digital Video などの安価でネットワーク接続可能な実時間メディア端末の出現により、その要求はますます高まっている。

連続メディアをサポートするネットワークシステムを設計する時は、以下のような問題を考えなければならない。まず、圧縮された連続メディアは時間的制約を持つ VBR の性質を示す

が、これはバーストを生じることや、自己相似性などの性質を持つことが知られている [8]。平滑化のために中間ノードにバッファを入れることも考えられるが、インタラクティブなアプリケーションではそれによって生じる遅延に耐えられないことが予想される。例えば、東京から米東海岸まで太平洋横断 ATM 回線を利用した専用回線上での RTT 値は約 170 msec との結果を得ている。広域ネットワークではこのように光速度の限界も無視できない。また CODEC や OS のオーバーヘッドなど、エンドホストでも遅延が生じる。これらを考えると、人間の知覚し得る約 200 msec 以下に合計の遅延を抑えるためには、ネットワークに遅延を挿入できるとは考えにくい。

VBR データのバースト性のもう一つの問題点として、ピーク値と平均値のデータレートの差が大きいことがあげられる。そのため、一本のストリームに対して一本の帯域幅を予約すると平均的な資源利用効率が低下する。バッファリングによる平滑化が現実には難しいため、効率のよい連続メディア通信を行なうためには現在のインターネットのように帯域を多数で共用するタイプのネットワークを用いて、品質を保ちながら通信を行なう枠組を実現しなければならない。

上記を背景とし、本論文では、インターネット上での連続メディア転送における問題のうち、特に last-one-mile 部分を取り上げ、その解決策の一つを提案する。last-one-mile に注目した理由は、バックボーンの高速化は指数関数的に進むと考えられるが、家庭と近傍のネットワークノードを直接接続する加入者線がコストやスケールメリットの理由からボトルネックになりつつあるためである。 そのため、加入者線に関して適切なネットワーク制御方法が確立されれば、長期的にもメリットが大きいと考えられる。

加入者線では、少数（一家族）の協調的なユーザーが一つの線を占有することができるものの、使える合計最大帯域幅は低く押えられている、という特徴を持つ。多人数で共用しなければならないが合計の帯域幅が大きいバックボーンとは対照的である。バックボーンにおいては、数多くのデータストリームが多重化されるため、大数の法則によるデータの平滑化がかなり期待できる。しかし、数本～数十本程度のストリームが多重化される程度の末端線であると、多重化効果を利用するだけでは不十分で、データストリームの優先度に基づく QoS 制御およびストリーム間調停を行なわなければならない。

このような QoS 制御は、汎用のアルゴリズムによって達成できるものではなく、ユーザーやアプリケーションプログラムの特性／利用パターンに深く依存する。例えば、テレビ電話で長話をしている息子のデータストリームよりも SOHO 環境で職場の同僚とテレビ会議をしている母親のストリームを優先したい、といった要求があるだろう。あるいは、異なるウィンドウに表示されているストリームのうち、前のウィンドウにあるストリームの品質を高く、後ろのウィンドウで隠れているストリームの品質を低くする、といった制御も考えられる。また I,P,B フレームといった異なる重要度をもつフレーム列から構成される MPEG ストリームと、すべて同じ重要度のフレームからなる M-JPEG ストリームでは、メディアスケーリングをする際の振舞いが異なる。すなわち、適応的な動的 QoS 制御が必要になり、かつそのポリシは利用者環境によって異なる。

以上を踏まえて、本論文では柔軟性のあるエンドホストのためのネットワークシステム VoR を提案する。VoR はアプリケーションプログラム、ミドルウェア、ネットワークドライバの協調によって QoS 変換・調停・保証を行なうメカニズムを提供する。QoS 変換・調停部分はメカニズムとポリシの分離の原則に基づいて設計されているため、カスタマイズが容易である。このシステムを RT-Mach および ATM ネットワークカードを利用して実装し、評価した。

本論文は以下のように構成されている。第 2 章では関連研究を紹介する。第 3 章で VoR の設計について述べる。第 4 章では実装プラットフォームについて述べ、第 5 章で実装の評価を行なう。第 6 章でまとめと今後の展望を述べる。

2 関連研究

Grossglauser[1] らは VBR トラフィックを効率的に転送する手法として RCBR (Renegotiated Constant Bit Rate) を提案している。RCBR では比較的短い時間間隔 (数秒から 10 秒程度) で CBR によって予約されたネットワークの帯域幅を変化させる。予約帯域の決定に関しては、蓄積メディアに適した計測ベースの手法と、ライブメディアのための線形予測の方法について論じている。RCBR では主にシーンの変化にともなう 1 秒 ～ 10 秒のオーダーのバースト性に着目し、それらを多重化によって効率的に吸収することを目的としている。また、総帯域幅の制約から予約帯域の変更ができなかった時は、それまで確保していた帯域幅を継続するようにすることで、最悪の場合でも緩い QoS 保証ができるようにしている。

本論文で提案する手法は、より短い時間 (数フレームなど) のバースト性も多重化によって吸収することを目的とする。RCBR ではシミュレーションによる評価のみであるが、本論文ではアプリケーションによるメディアスケーリングも視野に入れた柔軟性のあるシステム設計を行ない、実機上で実装、評価をしている。

長 [2] はルーターにトラフィック制御機能を実現するための汎用インタフェース ALTQ を FreeBSD 上に実装した。さらに、そのフレームワーク上に CBQ (Class Based Queueing) を実装し、評価を行なっている。CBQ などの帯域制御アルゴリズムの多くはシミュレーションによって評価されているに留まっていたが、PC ルーターの上でそれらのアルゴリズムが実現できるフレームワークが提供されたことで、実際の環境におけるテストができるようになった。ALTQ では異なるスケジューリングアルゴリズムを実現しやすいフレームワークは提供されているものの、動的な QoS の変更のためのインタフェースや機構は現在のところ用意されていない。しかし、VoR の動的 QoS 変更機能を ALTQ に組み込むことは可能であると考えており、今後の研究課題である。

3 VoR システム概要

本章では VoR ネットワークサブシステムの概要について述べる。VoR は 3 つの構成要素、ネットワークドライバ、ストリームコーディネータ、コーディネータインタフェースライブラリから構成される (図 1)。VoR は、あらかじめ設定された固定帯域幅に対して、同一のマシン上の複数のアプリケーションプログラムから生成される複数のストリームを QoS 変換・調停・保証をしながら多重化し、送出することを目的としている。そのため、ネットワークの帯域予約をする機能は含まれていない。

VoR にはネットワークドライバとストリームコーディネータが一つずつ、コーディネータインタフェースライブラリは各データストリームに一つずつ存在する。それぞれの機能の詳細については以下の各節で述べる。

アプリケーションプログラムはデータを生成し、ネットワークドライバにデータを書き込む。さらに、コーディネータインタフェースライブラリに対して QoS を指定する。コーディネータインタフェースライブラリはアプリケーションプログラムとストリームコーディネータのインタフェースを提供し、アプリケーション QoS からネットワーク QoS への変換ポリシーモジュールを組み込むことを可能にしている。 ここでネットワークの QoS に変換された指定は、ストリームコーディネータに渡される。ストリームコーディネータは異なるストリームの QoS 要求を調停し、ネットワークドライバのスケジューリングパラメータに変換する。この調停アルゴリズムもさまざまなものが考えられるので、ポリシモジュールとして組み込めるようになっている。ネットワークドライバにはスケジューラがあり、ストリームコーディネータによって設定されたパラメータによってスケジューリングを行なう。

VoR は緩い QoS 保証を行なう。例えば、ストリームコーディネータは QoS の調停計算を行なうが、これはアプリケーションプログラムの要求する QoS を全て満足するとは限らない。そのときはアプリケーションプログラムに通知が行なわれるが、データの送出はその時点

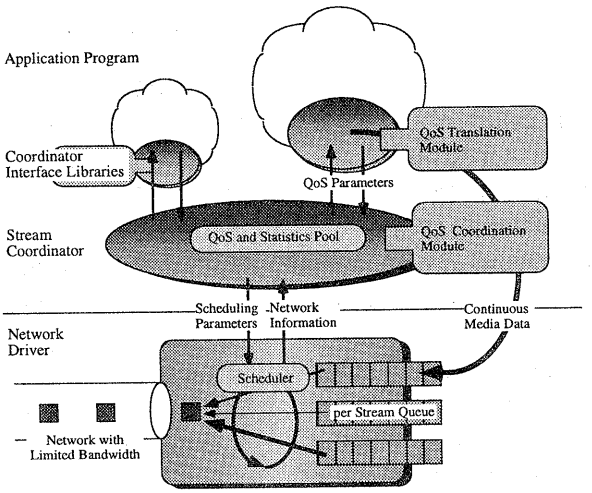

図 1: VoR システムの概要

での QoS パラメータを利用して行なわれる。あるいは、調停計算に要する時間、計算結果をネットワークドライバへ設定するタイミングによっては、QoS の変更がすぐにデータ転送に反映されないこともあり得る。このようなときはそれまで利用されていたパラメータがそのまま使われる。

3.1 ネットワークドライバ

既存のネットワークドライバに対しては、複数のストリームからのデータを QoS 指定に従って送出するためのスケジューラが加えた。受信側の機能に関しては通常のものと同じである。

ネットワークドライバはネットワークカードハードウェアがシェーピング機能を提供していることを仮定している。シェーピング機能が必要である理由は、ネットワークが指定する帯域幅を守る機能がネットワークカードに委ねられているためである。スケジューラは指定帯域幅内に指定の QoS に従ってデータを多重化するために存在し、それ自体では全体の転送レートを守ることはしない (図 2)。このような仮定は、インテリジェントなネットワークカードが増えている今日、妥当なものであると考えている。

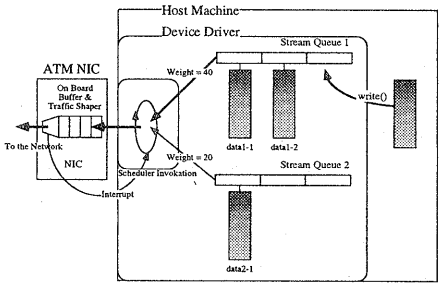

図 2: ネットワークドライバ内のデータスケジューリングについて

アプリケーションプログラムが `device_write()` (UNIX の `write()` に相当) によってドライバにデータを書き込むと、それらのデータはまずストリームごとのキューに保持される。ネットワークスケジューラはデータの DMA 転送が終了したときに返されるインタラプトをトリガーとして起動される。スケジューラが起動されると、その時点でネットワークに書き込める最大量 (通常はネットワークカード上のメモリの空き領域) を取得する。次に、その量と各ストリームごとの重みパラメータに従って、ネットワークへの送出量が決定される。重みパラメータと計算の優先度を後述するストリームコーディネー

タから変更可能にすることによって、QoSを動的に変化できるようになっている。

3.2 ストリームコーディネータ

ストリームコーディネータはユーザーレベルのミドルウェアとして実現されており、ユーザーからのQoS要求を調停し、ネットワークドライバの重みパラメータに変換して、設定を行なう。また、ネットワークドライバの統計情報を取得し、アプリケーションへのQoS変更要求を行なう。

ここで、以前に論じたように、ストリーム間のQoS調停は利用形態、適応範囲によってさまざまであり、これらを満たす汎用のアルゴリズムは存在しない。そこで、ストリームコーディネータではQoS調停計算モジュールへのインタフェースを定義し、モジュールを別途実現するようにした。QoS調停計算モジュールへのインタフェースとして、以下を定義した。

```
void DoArbitrationCalculation
   (QoSpool_t* poolp, schtbl_t* stp);
```

poolpによって全ストリームのQoS情報を渡す。計算の結果はスケジューリングパラメータテーブルであるstpに入れる。

3.3 コーディネータインタフェースライブラリ

コーディネータインタフェースライブラリはアプリケーションプログラムに対するインタフェースを提供する。重要な機能は2つある。一つはアプリケーションが指定するQoSをネットワークが理解するQoSに変換するものである。例えば、動画ではフレーム／秒、ピクセル数、色のビット数、といったパラメータでQoSを指定する。これをストリームコーディネータで異種のストリームを調停をするできるように、平均ビットレート、ピークレート、最大遅延時間といったパラメータに変換してやらなければならない。このような変換はメディアの種類、フォーマットなどによって異なるので、これらの変換アルゴリズムも外部のモジュールとして実現できるようにし、システムの柔軟性を高めている。変換モジュールへのインタフェースは次の2つである。

```
SPtoNP(np_t* np, void* userQoSp);
/* アプリケーションQoSをネットワークQoS
に */
NPtoSP(void* userQoSp, np_t* np);
/* ネットワークQoSをアプリケーションQoS
に */
```

もう一つの機能は、ストリームコーディネータによる調停計算の結果、アプリケーションに対してQoS変更の要求を出すことが決まった時に、通知を行なうものである。これはストリームコーディネータとコーディネータインタフェースライブラリの共有変数を介して行なう。

ストリームコーディネータがQoS変更を決めると、共有変数にフラグを立てる。アプリケーションはコーディネータインタフェースライブラリを介して定期的にこの変数をポーリングしなければならない。メディアスケーリングなどの処理はそれほど緊急性がないので、このようにすることでプログラミングを簡単にしている。フラグが立ったら、ストリームコーディネータから指定のQoSパラメータを取得する。それに従ってアプリケーションはメディアスケーリングを行ない、新しいQoSをセットする。

QoSを理解しないアプリケーションはこの通知を無視することも可能である。その場合でもストリームコーディネータによって資源割り当ては変更されるので、少ない資源に対して無駄にデータを流すことになる。すなわち、品質低下が起こりやすくなる。このようにしてQoSを扱うことをアプリケーションに促す。

4 実装

前章で説明した設計を慶応大学MKngプロジェクトで開発したRT-Machの上で実装した。ネットワークカードとして、Efficient NetworksのENI-155pを利用した。このカードはオンボードのメモリ上でAAL5のSAR (送信時にAAL5 PDUをセルに分割する、および受信時にセルからAAL5 PDUを再構成する) 機能を

持っている。そのため、ドライバ内部で可変長の PDU を構成し、それをカードに転送すればセルとして送出される。現在のところ、実装上の理由から最大 4096 byte のパケットを構成するようになっている。

現在の実装では、帯域制限をするために ENI-ATM カードに提供されているトラフィックシェーピング機構を利用している。これはレジスタに帯域幅を指定すると、ホード上のシェーパーが指定以上のトラフィックを流さないように制御するものである。このシェーパーは VCI の集合に対して適用することができる。

デバイスドライバはワシントン大学で開発された FreeBSD 用ドライバを MKng プロジェクトで RT-Mach に移植したものに変更を加えた。現在はシステム全体が C 言語と MIG (Mach Interface Generator) を利用して記述されている。

5 実験による評価

実装したシステムの基本的な特性を調べるために、次の計測を行なった。実験は、すべて 10 Mbps の帯域幅を指定しておこなった。実験環境として、Pentium II 300 MHz の CPU を持つ PC 2 台を ATM スイッチににによって対向に接続した。

5.1 スケジューラの起動間隔

まず、スケジューラの起動間隔を調べた。現在の実装では、カードからの AAL5 パケット転送終了のインタラプトをトリガとしてドライバのスケジューラが起動されている。実際の連続メディアデータを長し、その間隔を調べた。時刻は Pentium プロセッサの機能として提供されている、現在のクロックの値を 64 bit で返すレジスタ (Pentium Cycle Counter) にアクセスすることで計測した。

転送するデータとして、CNN ニュースおよび間のコマーシャルを 30frame/sec でデジタイズし、M-JPEG 圧縮したもののトレースを利用した (図 3)。これを RT-Mach のリアルタイムスレッドを利用して 30msec ごとにドライバに渡した。

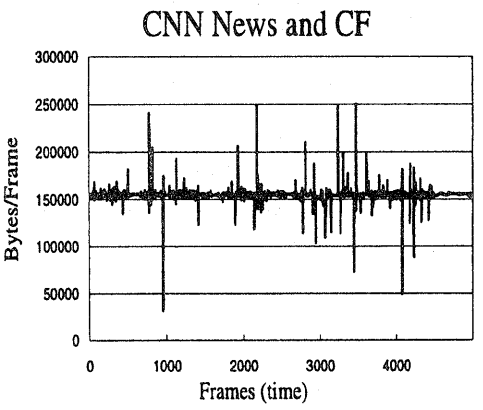

図 3: 実験に利用した M-JPEG のトレースの一部、CNN ニュースおよび CM

この実験の結果の一部は 図 4 に示されている。全体を通じて、ほぼ一定の間隔でスケジューラが起動されていた。これは、可変長のデータ (平均 15000bytes) が最大 4096 byte の AAL5 PDU に分割されており、それぞれのパケットの DMA 転送に対してインタラプトが発生しているためである。インタラプト間隔が広い部分は、一つのフレームの送出が終了し、次のフレームの到着を待っているアイドルの状態である。動画像を転送する際の時間的制約は通常数十ミリ秒のオーダーであり、ネットワークからのインタラプトは数ミリ秒程度の間隔で発生することを考えると、インタラプトをトリガーとしてスケジューラを起動する方式によってうまくスケジューリングができると考えられる。

5.2 多重化時の送出の様子

次に、最大 5 本のストリームを重みを付けながら多重化したときの特性を調べた。33msec ごとに 4000 byte のパケットを 4 回 (約 4Mbps) 送出する CBR ストリームを用いた。帯域は静的に 40, 30, 20, 5, 5 の重みづけをし、ストリームが加わることによって、実際の帯域割当量が変化していく様子を調べた。

トラフィックの測定は、マシンが接続された

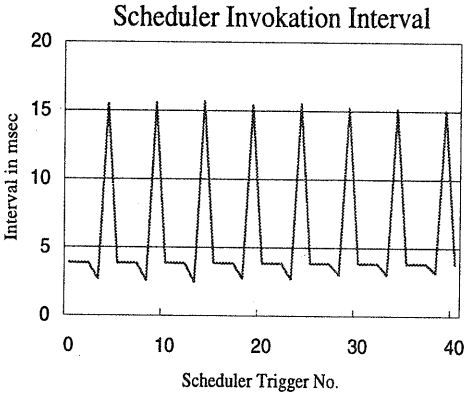

図 4: インタラプトをトリガとするスケジューラの起動間隔

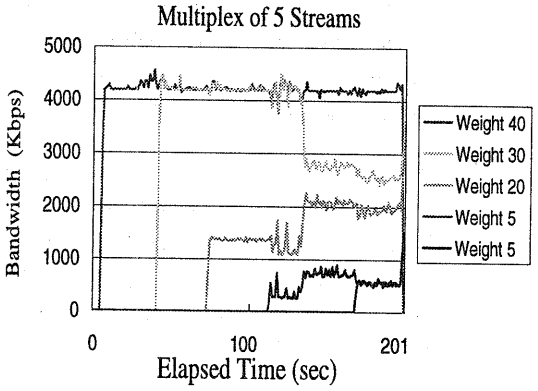

図 5: 最大 5 本のストリームを多重化したときの実際の帯域

ATM スイッチを監視するトラフィックモニタリングソフトウェア (Fore 社 ForeView) を利用した。

結果は図5のようになった。2本目まではストリームの合計が全体帯域幅を越えていないために、フルスピードで送出が行なわれている。3本めのストリームが開始されてから、ストリームの合計が全体の帯域 10Mbps を越える。ところが、3本目が始まった直後では、新たなストリームに対して帯域が重み通りに割り当てられなかった。4本目が開始され、、過渡的に不安定な状態を越えて、定常状態になっている。定常状態では、ほぼ指定帯域が守られているが、Weight 40 のストリームの実際の送出量が若干高め、Weight 30 のストリームが、重みによって指定されている帯域よりも送出量が少なめである。このような過渡的状態や帯域割当の不正確さが発生する理由と改善方法について検討を行なっている。

6 まとめと今後の課題

限定された帯域下で効率的に連続メディアを扱うためのネットワークシステムとして、VoR を提案し、その実験評価を示した。現在、映画、ニュース、ホームビデオ、講義など、さまざまなコンテンツの連続メディアストリームを作成し、その特性を調べている。このことにより、どのような QoS 調停アルゴリズム、メディアスケーリングの手法が有効であるかを検討し、今回提案したシステムの上にそれらを実現していく予定である。

参考文献

[1] Grossglauser, M., Keshav, S., Tse, D.N.C., "RCBR: A Simple and Efficient Service for Multiple Time-Scale Traffic", *IEEE/ACM Transactions on Networking, vol.5, No.6*, December, 1997

[2] "PC Unix ルータによるトラフィック制御の実現", In Proceedings of *Internet Conference '97*, December 1997.

[3] Matsui, Y., Tokuda, H., "VOR: A Network System Framework for VBR over Reserved Bandwidth", In Proceedings of *4th International Workshop on Interactive Distributed Multimedia Systems and Telecommunication Services* (LNCS 1309), September, 1997

[4] Clark, D.D., Tennenhouse, D.L., "Architectural Considerations for a New Generation of Protocols", In *Proceedings of ACM SIGCOMM '90*, 1990.

[5] Delgrossi, L., Halstrick, C., Hehmann, D., Herrtwich, R., Krone, O., Sandvoss, J., Vogt, C., "Media Scaling for Audiovisual Communication with the Heidelberg Transport System", In *Proceedings of ACM Multimedia '93*, 1994.

[6] Kiczales, G., "Towards a New Model of Abstraction in Software Engineering", In *Proceedings of the IMSA '92 Workshop on Reflection and Meta-level Architecture*, 1992.

[7] Tokuda, H., Nakajima, T., Rao, P., "Real-Time Mach: Towards a Predictable Real-Time System", In *Proceedings of USENIX First Mach Symposium*, 1990.

[8] Garrett, M.W., Willinger, W., "Analysis, Modeling and Generation of Self-Similar VBR Video Traffic", In *Proceedings of ACM SIGCOMM '94*, 1994.