システムソフトウェアと
オペレーティング・システム 85－3
（2000．8．4）

マルチメディア通信における帯域予約型資源管理方式の開発

間島 敦史† 毛利 公一†† 吉澤 康文††

†東京農工大学大学院工学研究科
††東京農工大学工学部

ネットワーク上で提供されるサービスは，リアルタイム性を保証する必要の無いものと，リアルタイム性を保証する必要のあるものに分類することができる．しかし，既存のオペレーティングシステムではリアルタイム性を保証する通信機構が提供されていない．我々は，これらの問題を解決するために，ソケットを単位とした，予約型帯域保証機構を構築した．本論文では，予約型帯域保証機構の構成とその評価について述べる．本機構を用いることによって，リアルタイム性を保証することが可能となる．また，ソケットを単位として帯域保証を行うことで，1つのアプリケーションが，複数の異なるQuality of Serviceに基づく通信を，回線の種別に依存することなく，同時に満たすことが可能となる．

Development of Resource Reservation Mechanism for Multimedia Communication

Atsushi Majima† atm@ymbll.ei.tuat.ac.jp
Koichi Mouri†† mouri@lavender.org
Yasufumi Yoshizawa†† yoshiza@cc.tuat.ac.jp

†Graduate School of Engineering,
Tokyo University of Agriculture and Technology
††Faculty of Engineering,
Tokyo University of Agriculture and Technology

Services provided on the network consist of two types. One requires to guarantee real-time communication. The other doesn't require to guarantee. However, current operating systems don't have the mechanism to guarantee the real-time communication. To solve this problem, we have been developed the bandwidth-guarantee mechanism based on reservation for each socket. In this paper, the structure and the evaluation of the mechanism are described. The mechanism achieves to guarantee the real-time communication. A application can use plural communications founded on each Quality of Service simultaneously by guarantee the bandwidth for each socket.

1 はじめに

現在，パーソナルコンピュータの低価格化と情報インフラの急速な発展に伴い，ネットワークを利用した様々なサービスが提供されている．それらのサービスは，FTPなどのようにリアルタイム性を保証する必要の無いものと，VOD(Video on Demand)システムのようにリアルタイム性を保証する必要のあるものに分類することができる．しかし，既存のオペレーティングシステムではリアルタイム性を保証する通信機構が提供されていないため，VODシステムを構築することが困難となっている．この問題を解決するための一つの手法として，予約方式による帯域保証のための機構が注目されている[1]〜[4]．我々は，特にソケットを単位とした，予約型帯域保証機構(以下，帯域保証機構と記す)を構築した．

本機構によって帯域を保証することにより，通信量の変動が軽減されるため，リアルタイム性を保証することが可能となる．また，ソケットを単位として帯域保証を行うことで，1つのアプリケーションが，複数の異なるQoS(Quality of Service)に基づく通信を，同時に満たすことが可能となる．さらに，回線の種別に依存しない実装となっているため，イーサネットなどの帯域を保証するための機能を持たないネットワークにも適用可能となっている．

以下，本論文では，2章で帯域保証方式について述べ，3章で帯域保証機構の全体構成について述べる．4章では，本機構の中心となるデータ送信マネージャの実装について述べ，5章で帯域保証機構が提供するインタフェースについて述べる．6章では，本機構の性能評価について述べる．

2 帯域保証方式

マルチメディア通信においては，動画像や音声といったデータが送信される．また，それらのデータ送受信の制御のための通信も必要となる．すなわち，1つのアプリケーション内で，必要とする帯域が異なる，複数の通信が行われる．本機構では，これらを可能とするために，帯域保証の単位をソケットとしている．

また，ATMと同様に，ソケット毎に専用の仮想回線を提供する方式を用いる．この仮想回線は，次に示すようなフロー制御によって実現している．

(1) アプリケーションが送信を要求したデータを受け取る．

(2) アプリケーションが予約した帯域に合わせ，一定時間毎に(1)のデータを適切なサイズに区切り，パケットをネットワークに流す．

例として，アプリケーションが帯域nバイト/秒を予約した場合，これは，t秒毎にデータをsバイト送信することによって実現することができる．ただし，$n = s/t$とする．

(1) アプリケーションがpバイト送信する．

(2) 一度帯域保証機構内でデータを保存する．

(3) そのサイズに応じて送信するデータのサイズを決定する．

  (a) $p \leq s$ならば，pバイトのデータ全てを送信する．

  (b) $p > s$ならば，先頭のsバイトだけ送信する．

(4) 残りの$p-s$バイトはt秒後に(3)の手順を用いて再度送信する．

上記の方式を用いることによって，アプリケーションの通信に対する帯域が確保され，サーバとクライアント間のデータ送受信のリアルタイム性を保証する．ただし，複数のネットワークセグメントをまたがる通信においては，それらを接続するルータにおいてRSVPなどの帯域保証のための機構を利用することにより帯域保証が可能となる．

3 予約型帯域保証機構の全体構成

帯域保証機構の全体構成を図1に示す．プロセスでは，他のプロセスが必要とする帯域を知ることができないため，本機構はカーネル内に実装されている．内部は，データ送信マネージャ，インタフェース，帯域予約テーブル，データプールから構成される．

帯域予約テーブルは，ソケット記述子とそのソケットが予約する帯域の組が保持される．データプールは，アプリケーションが送信したデータを一時的に保持しておく領域である．送信データは，ソケット記述子毎にリスト構造で格納される．データ送信マネージャは，帯域予約テーブルの管理，データプールの管理，帯域を保証しながらのデータ送信を行う．インタフェースはユーザに対して提供されており，ユーザはこのインタフェースを用いて帯域の予約等の処理を行う．

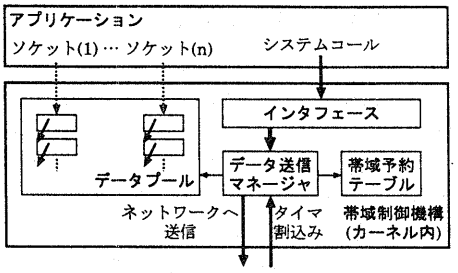

図 1　予約型帯域保証機構の全体構成

以上の構成により，アプリケーションが送信するデータを，一度帯域保証機構内にコピーする．さらに，タイマ割込みを利用して定期的にデータを送信することによって，帯域を保証する．

4　データ送信マネージャ

データ送信マネージャは，次に示す処理を行う．

- アプリケーションからの帯域予約および帯域解放要求の処理
- アプリケーションによる，データ送信要求の処理
- 一定周期毎の，データ送信処理
- ログ取得

本章では，以下，上記の各処理のためのデータ構造と機構について述べる．

4.1　データプールと帯域予約テーブル

データプールと帯域予約テーブルは，その内容の操作のための検索の効率を向上させるために，図2に示す bwp_struct 構造体を用いてプロセス毎に管理している．bwp_struct は，プロセスID，ソケット単位のデータプールと帯域予約テーブルへのポインタを管理している．このポインタは，同時にオープンできるファイル記述子の数である，NR_OPEN だけ用意される．NR_OPEN は，Linux のヘッダファイルでは，256 と定められている．

ソケット単位のデータプールと帯域予約テーブルは，図 3 で示す bwc_struct 構造体で管理される．bwc_struct は，プロトコル，帯域幅，プロセス構造体へのポインタ，およびデータプールと帯域予約テーブルを保持する．リスト構造を構成するためのメンバ

```
struct bwp_struct {
  bwp_struct *next;
  pid_t       pid;
  bwc_struct *bwc[NR_OPEN];
};
```

図 2　bwp_struct 構造体

```
bwc_struct {
  bwc_struct          *next;      // データ転送リスト
  int                 type;       // 型(UDP/TCP)
  struct BandWidth    bw;         // 帯域幅
  struct task_struct *bwc_task;
  struct bwcbuf       *bw_head;   // 送信データの先頭
  struct bwcbuf       *bw_tail;   // 送信データの最後
};
```

図 3　bwc_struct 構造体

```
struct bwcbuf {
  struct bwcbuf *next;
  char          *buf;     // バッファ
  int           bufsize; // バッファサイズ
  int           curr;     // データポインタ
};
```

図 4　bwcbuf 構造体

```
struct BandWidth {
  int bandwidth; // 予約された帯域
  int s;         // 単位時間当りの送信データサイズ
};
```

図 5　BandWidth 構造体

next は，データプールの内容が空でないソケットをリンクする．これは，データ送信処理の際の検索効率を向上するために使用される．プロセス構造体は，ソケットのファイル構造体，i_node 構造体を参照するために必要となる．i_node 構造体は共用体で構成され，この変数に socket 構造体へのポインタが含まれる [5]．

データプールは，図4で示される bwcbuf 構造体で管理される．bwcbuf 構造体では，バッファとそのサイズ，バッファ内において次に送信すべきデータを示すためのポインタが保持される．

帯域予約テーブルは，図5で示される BandWidth 構造体で管理される．BandWidth 構造体では，アプリケーションが予約した帯域と，定期的に送信されるバイト数が格納される．このバイト数は，2章で述べた帯域保証方式の s を意味する．

4.2 帯域予約と帯域解放

アプリケーションから帯域予約システムコールが発行されると，データ送信マネージャは，割当て可能な帯域の有無を確認する．割当てが可能な場合，アプリケーションのプロセスIDとファイル記述子に基づいて，情報を格納するための bwp_struct 構造体，bwc_struct 構造体を生成する．さらに，BandWidth 構造体の内容を設定する．このように，bwp_struct と bwc_struct は，on demand で生成される．この帯域の予約は，アプリケーションにおいて socket()[6] を呼び出した後に用いる．

帯域解放システムコールが発行されると，帯域予約システムコール発行時に生成された構造体を破棄する．ただし，bwp_struct は，自身に属する bwc_struct が全て存在しなくなるか，アプリケーションが終了するまで破棄されることはない．

4.3 データ送信要求

アプリケーションが write() や sendto() などのデータ送信要求を実行することにより呼出される．データ送信要求が来ると，データ送信マネージャは，データを保持するためにバッファを割り当て，バッファにデータをコピーする．さらに，bwc_struct のメンバである wc_tail に連結する．また，bwc_struct のメンバである next を連結する．これによって送信データを持つ bwc_struct の検索が容易となる．

データの送信要求では，アプリケーションの送信データをコピーする処理のみを行う．実際の送信は，4.4節で述べるデータ送信処理内で行われる．

4.4 データ送信

データ送信処理は，タイマ割込みを用いて定期的に実行される．データ送信処理が呼出されると，データ送信マネージャは，bwc_struct のメンバである next を用い，送信すべきデータのある bwc_struct を検索する．バッファ内の，bwcbuf 構造体のメンバ curr で示される位置を先頭とし，BandWidth 構造体のメンバ s で示されるサイズだけネットワーク上に送信する．

現在，MPEG ファイルが 30 フレーム/秒 [7][8] であることを考慮し，1/30 秒毎にデータ送信を行っている．

4.5 ログ取得の開始と終了

アプリケーションから，ログ取得開始システムコールが発行されると，データ送信マネージャは，ロギングフラグをオンにする．これによって，データ送信処理が行われた際に，プロセスID，ソケット記述子，送信したデータのサイズ，送信時刻を記録される．

ログ取得終了システムコールが発行されると，ロギングフラグをオフにする．さらに，アプリケーションが持つメモリ領域に，指定された個数のログをコピーする．

5 インタフェース

データスプールマネージャがアプリケーションに対して提供するインタフェースを次に示す．

(1) 帯域予約システムコール

書式: `int setbandwidth(int sockfd, int type, int bandwidth)`

アプリケーションが帯域を予約するために用いるインタフェース．アプリケーションは，socket() システムコールを呼び出した後，本インタフェースを用いて必要な帯域 (バイト/秒) を帯域制御機構に通知する．アプリケーションは，帯域を予約するソケットのソケット記述子，ソケットの種類，必要な帯域 (バイト/秒) をカーネルに宣言する．成功した場合正の値が返される．

(2) 帯域解放システムコール

書式: `int clearbandwidth(int sockfd)`

ソケットに割り当てられた帯域を解放する．失敗すると負の値が，成功すると sockfd が返される．

(3) ログ取得開始システムコール

書式: `int startbwcdmtr()`

帯域制御機構がデータ送信を行った際に，そのプロセスID，ソケット記述子，送信したデータのサイズ，その時刻を帯域制御機構内のログバッファに記録するためのインタフェース．

(4) ログ取得終了システムコール

書式: `int quitbwcdmtr(mtr_struct *to, int num)`

表 1　サーバに用いた PC の仕様

CPU	Intel Pentium Pro 200MHz
メモリ	64MB
ネットワーク	10Mbps
OS	Linux 2.0.34
再生ソフト	MPEG_PLAY Ver. 2.0
送信データサイズ	10,832,624 バイト

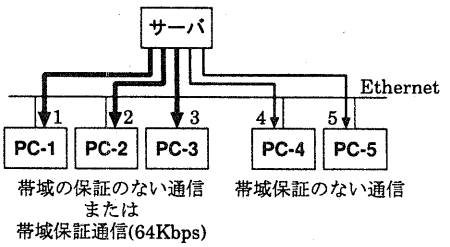

図 6　実験環境

帯域制御機構内のログバッファから，ユーザ空間にデータをコピーするためのインタフェース．to は，ユーザ空間におけるコピー先のアドレス，num はコピーするデータの個数である．ユーザプロセスが指定した個数と，カーネルが記録したデータの個数を比較し，小さいほうに合わせてコピーする．

6　性能評価

6.1　評価方法

評価は，表 1に示す仕様の PC を帯域保証機構を適用したサーバとし，5 台のクライアント PC とともに，図 6に示すネットワーク環境で行った．このとき，サーバから，図 6に示すように，クライアント PC1 台毎に 1 つのソケットを生成した．

上記の環境において，次の 2 つの計測を行い，帯域の保証機構が行われていることを示す．

(1) 帯域を保証していない通信を行った場合のトラフィック (表 2の計測 1)

(2) 帯域を保証した通信と保証しない通信を同時に行った場合のトラフィック (表 2の計測 2)

5 つのソケットでは，全て MPEG1 Movie ファイル (約 10MB) を送信した．ただし，それぞれで用いるプロトコルは，1〜3 が UDP，4〜5 が TCP(FTP) となっている．

表 2　ソケットの割当て

番号	クライアント	割当て帯域 (KB/s)	
		計測 1	計測 2
1	PC-1	なし	64
2	PC-2	なし	64
3	PC-3	なし	64
4	PC-4	なし	なし
5	PC-5	なし	なし

6.2　計測結果と評価

実験では，文献 [9] で挙げられているトラフィックキャプチャとトラフィックアナライザを用い，1 秒毎の各ソケットにおける送信データ量を計測した．

帯域を保証していない通信を行った場合のトラフィックの計測結果を図 7に示す．図 7より，帯域を予約せずに送信を行った場合，ソケット 4 と 5 によってネットワークに負荷をかけたため，ソケット 1〜3 のデータがほとんど送信されていない．帯域を割り当てずに送信を行った場合，最初の数秒しか送信されていない．

帯域を保証した通信と保証しない通信を同時に行った場合のトラフィックの計測結果を図 8に示す．図 8より帯域保証機構を用いることにより，ソケット 1〜3 の帯域が保証され，MPEG1 MOVIE ファイルの送信が行われていることがわかる．FTP によるネットワークの負荷の影響が，帯域を予約せずに送信した場合より小さい．FTP の送信を開始した直後は多少不安定になるが，それを除くと，予約前に比べて大幅に安定して送信することが可能となっている．ただし，割込みの影響により，データの送信量が減少すると，割込みにあわせて，次の時刻には送信量を増やしている．この結果から帯域保証が可能であることがわかる．

7　おわりに

本論文では，我々が Linux に構築した予約型帯域保証機構の構成と評価について述べた．本機構によって，通信におけるリアルタイム性を保証することが可能となる．また，ソケットを単位として帯域保証を行うことで，1 つのアプリケーションが，複数の異なる QoS に基づく通信を，回線の種別に依存することなく，同時に満たすことが可能となった．

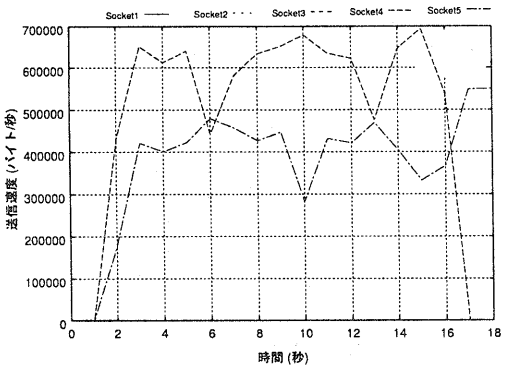

図 7　帯域を保証しない場合

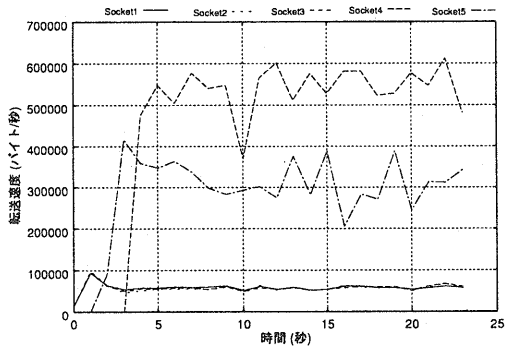

図 8　帯域を保証した場合

性能評価では，マルチメディアデータの送信実験を行った．帯域予約を行わない場合は，負荷によりデータの送信速度に乱れがあった．しかし，予約型帯域保証機構を用いた場合は，その帯域が保証され，データが送信されていることを確認した．

今後の課題として，ネットワークセグメントをまたがるような通信においても帯域を保証することが可能とする機構を実装することが挙げられる．また，実際に VOD システムに応用する予定である．

参考文献

[1] 長 健二朗, "インターネットのトラフィック制御, " 情報処理, Vol. 40, No. 10, pp. 1014–1019, 情報処理学会, 1999.

[2] 長 健二朗, "*PC Unix* ルータによるトラフィック制御の実現, " インターネットコンファレンス '97 論文集, JUS, 1997.

[3] Peter Druschel, "*Operating system support for high-speed communication: techniques to eliminate processing bottlenecks in high-speed netowrking are presented,* " Communications of the ACM, Vol. 39, No. 9, pp. 41–51, 1996.

[4] Ion Stoica, Hui Zhang, "*A Hierarchical Fair Service Curve Algorithm for Link-Sharing, Real-Time and Priority Service,* " Proceedings of the ACM SIGCOMM Conference: Applications, Technologies, Architectures, and Protocols for Computer Communication(SIGCOMM-97), Computer Communication Review, Vol. 27,4, pp. 249–262, ACM Press, 1997.

[5] R. Card, E. Dumas, and F. Mevel, "*Linux 2.0* カーネルブック, " オーム社, 1999.

[6] W・リチャード・スティーヴンス 著, 篠田 陽一 訳, "*UNIX* ネットワークプログラミング, " トッパン, 1997.

[7] 藤原 博, "実践 *MPEG* 教科書, " アスキー, 1995.

[8] 映像情報メディア学会, "ディジタルメディア規格ガイドブック, " オーム社, 1999.

[9] 毛利 公一, 趙 強, 吉澤 康文, "ネットワーク機器の分散配置によるトラフィック軽減手法, " マルチメディア・分散・協調とモバイル (DICOMO 2000) シンポジウム論文集, pp. 403–408, 2000.