マルチメディア通信と分散処理　84－3
（１９９７．９．１２）

アイソクロナス送信制御によるEthernet上でのQoS保証

岩嵜正明，中野隆裕，竹内　理，中原雅彦
（株）日立製作所システム開発研究所

複数のリアルタイム・スレッドが動作する環境において、数百マイクロ秒以下のスケジューリング遅延時間を保証した高精度周期駆動を実現するアイソクロナス・スケジューリング技術、および、これを利用したトラフィック・シェーピング技術(ITM)を開発した。これらの技術とEthernet上での帯域割り当て制御を実現するトラフィック制御プロトコル（TTCP）を組み合わせることにより、FTP等の非リアルタイム・トラフィックが混在する高トラフィック状態のEthernet上において、高品質なビデオ・データ転送が可能となることを実験で確認した。

QoS Assurance on an Ethernet using Isochronous Traffic Shaping

Masaaki Iwasaki, Takahiro Nakano, Tadashi Takeuchi, Masahiko Nakahara.
Systems Development Laboratory, Hitachi, Ltd.

We have developed an isochronous scheduling scheme that enables accurately periodic executions of real-time threads with the jitters less than few hundred micro-seconds. Besides, we have invented a traffic shaping scheme (ITM) using the scheduling scheme and a new traffic control protocol (TTCP) that enables the bandwidth admission for an ethernet. We have confirmed that these technologies can achieve the high-quality real-time video-data transfer on an ethernet segment, while both real-time traffic and non real-time traffic coexist on it.

１．はじめに

インターネット上でのビデオ･データのリアルタイム転送に不可欠なデジタル･ビデオ処理技術は急速に進歩している。現在、広くLANに普及しているEthernetの物理的バンド幅は10Mbpsないし100Mbpsであり、原理的には、MPEG形式等に圧縮されたビデオ・データを転送可能である。

しかしながら、実用に耐える高品質なビデオ・データ通信が可能なコンピュータ・ネットワーク・システムを実現するには、End-to-EndでのQoS (Quality of Service) 保証の実現を始め、解決すべき技術課題が多い。既存のOSに組み込まれているスケジューリング・アルゴリズムや通信プロトコルは、連続メディア・アプリケーションに不可欠なQoS保証機能を欠いている。

我々は、ビデオ・データの高品質リアルタイム転送の実現に向けて、マイクロカーネルTactixの研究開発を進めている。本稿では、連続メディア・アプリケーション向けのリアルタイム・スケジューリング技術と、これを応用したリアルタイム通信技術について述べる。

２．リアルタイム・スケジューリング

ここでは、まず、既存のリアルタイム・スケジューリング技術を連続メディア･アプリケーションに適用する場合の限界について述べる。次に、新たに開発したアイソクロナス・スケジューリング方式について説明する。

2.1. 既存スケジューリング技術の限界

ビデオ・データを扱う連続メディア・アプリケーションの多くは、一定周期毎に画面への描画処理を行う。この周期は、毎秒何フレームを描画するかによって決まる。一般に、連続メディア・アプリケーションでは、周期的にプログラムを実行するスケジューリング機能が要求される。

周期的なスケジューリング機能を提供するアルゴリズムとして最も有名なのは、Rate Monotonic Scheduling方式である。しかしながら、この方式は、一周期毎に各ス

レッドが必要とするCPU時間が厳密に一定でなければ、各スレッドの周期実行を保証できない。このため、外部割り込みや非周期的なスレッドが存在する現実のシステムでは、スケジューリングの遅延時間が大きくなってしまう。また、低優先度スレッドが共有資源を占有しているために、高優先度スレッドの実行が遅延する優先度逆転問題が発生する。

これらの問題点の解決を目的として、Earliest Deadline First Algorithm, Priority-Ceiling Protocol, Deferrable Server Algorithm, Processor Capacity Reservation等の方式が提案されてきた。Rate Monotonic方式が、複数の周期スレッドを静的にスケジューリングするのに対し、これらの方式は、時間的な制約条件の優先度を動的に判定し、スケジューリングすべきスレッドを決定する。

しかしながら、これらの動的なスケジューリング方式は、以下の様な問題点を有する。

(a) 時間的な制約条件の優先度、例えば、どのスレッドがデッドラインに最も近いか、あるいは、どのスレッドが共有資源をロック中かといった状態スキャンのために、スケジューリング・コストが増大する。

(b) 多数のスレッドが存在する場合、それらの時間的制約条件を、同時に、すべて満足することは必ずしも可能ではない。

(c) あるスレッドが、割当時間を超過してプロセッサを使用し続けると、他のスレッドがその影響を受け、デッドライン・ミスを連鎖的に発生させてしまう。

これらの問題のために、既存の方式では、リアルタイム・スレッドが多数存在する場合やプロセッサ負荷が高い場合には、スケジューリング遅延が増大してしまう。

2.2. アイソクロナス・スケジューリング方式

アイソクロナス・スケジューリングは、連続メディア処理において典型的な周期的動作を必要とするリアルタイム・スレッド（周期スレッド）を、正確に周期駆動させる機能を提供する。アイソクロナス・スケジューリングは、複数の周期スレッドと複数の非周期スレッドとが混在する環境下で、各周期スレッドのスケジューリング遅延時間を数百マイクロ秒以内に抑えることができる。また、同時に、突発的なイベントに対しても、数百マイクロ秒以下の遅延時間で応答できる[1-3]。

アイソクロナス・スケジューリング方式は、各周期スレッドの実行時間を静的に予約する段階と、この予約情報に従って、タイマ割り込みを契機にスレッドを動的にディスパッチする段階とから構成する。

2.2.1. タイムスロットの予約

アイソクロナス・スケジューリング方式では、アプリケーションに各スレッド毎の「周期」と「1周期内のCPU時間」を宣言させる。スケジューラは、これらの情報から、周期実行の開始前にタイムスロット・テーブルを生成する。図1に示す様に、タイムスロット・テーブルは、CPU時間を等間隔のタイムスロットに分割し、各タイムスロット毎に最優先実行する周期スレッドを割当てたCPU時間予約表である。尚、ひとつのタイムスロットの長さは、タイマ割り込み間隔の十分の1程度である（デフォルトでは、タイマ割り込み間隔を10ミリ秒、タイムスロット長を1ミリ秒に設定している）。

各スレッドはタイマ割込み間隔を単位として周期を指定できる。また、タイムスロット長を単位として1周期内のCPU時間を指定できる。タイムスロット・テーブルの大きさは、各スレッドが指定した周期の最小公倍数に比例する。実装上は、指定できる周期をタイマ割込み間隔の2の冪乗倍に制約し、このテーブルの大きさを抑えている。また、アプリケーションによっては、1周期毎に必要なCPU時間が変動する。この様な場合には、アプリケーションに1周期内のCPU時間の上限値を指定させ、各周期毎に余剰となったCPU時間を解放させることができる。

スケジューラは、関数start-cyclic-execの発行により、スレッドが周期駆動モードに遷移する時、即ち、周期スレッドが追加される都度、タイムスロット・テーブルを再構築する。タイムスロット・テーブルの再構築処理では、周期が短いスレッドから順にタイムスロットを割当てる。周期が長く、かつ、1周期に必要なCPU時間のタイムスロットが連続して割当てられない場合、複数の不連続な空きタイムスロットを割当てる。

2.2.2. タイマ駆動ディスパッチ

アイソクロナス・スケジューリング方式では、原則的に、タイマ割込みを契機にディスパッチャを起動する。ディスパッチャは、前記のタイムスロット・テーブルを参照し、各タイムスロットにおいて最優先実行すべき周期スレッドを最優先度状態（Raised状態）に遷移させ、プロセッサにディスパッチする。

図1に示す様に、各タイムスロットにおいて、最優先実行される周期スレッドは高々ひとつであり、Raised状態の周期スレッド以外の周期スレッドは、当該タイムスロット内ではDepressed状態となる。Raised状態の周期スレッドは、割込みハンドラを除いて、最優先に実行される。一方、Depressed状態の周期スレッドは、時間が経過して割当てタイムスロットに到達し、Raised状態に遷移しない限りディスパッチされない。

タイマ割り込みの発生以外に、周期スレッドが能動的にプロセッサを解放した場合、あるいは、割り当てCPU時間の超過を検出した場合にも、ディスパッチャが起動される。例えば、実行時に1周期内のCPU時間が余った場合、Raised状態の周期スレッドは、関数thread-dispatchの発行により、能動的にDepressed状態に遷移し、

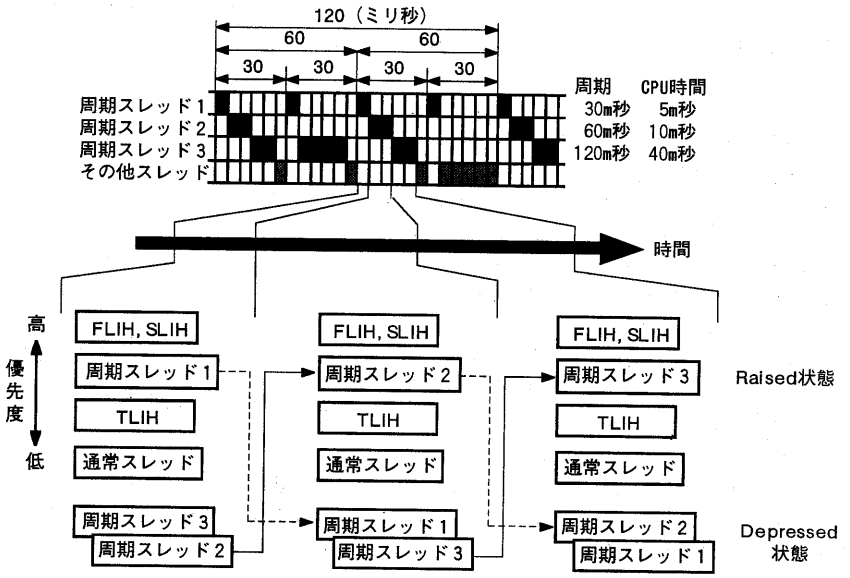

図1. タイムスロット・テーブルとタイマ同期ディスパッチの概要

余剰なCPU時間を解放することができる。この余剰CPU時間は、他の周期スレッドに割り当てられることはなく、非周期的なスレッドの実行に割当てられる。これにより、1周期内のプロセッサ使用時間の変動が、他の周期スレッドの起動時刻に影響することを防ぐことができる。また、非周期的なスレッドに割り当てるCPU時間を増加させることで、突発的なイベント発生への平均応答遅延時間を短縮することができる。

3．リアルタイム通信技術

ここでは、まず、Ethernet上のリアルタイム通信実現に向けた技術課題について述べる。次に、Ethernetにおけるトラフィック量と遅延時間の関係を定量的に分析する。最後に、既存の通信プロトコルやハードウェアを変更せずに、Ethernet上でパケット転送遅延時間を保証するトラフィック・シェーピング技術およびトラフィック制御プロトコルについて説明する。

3.1. 既存の通信プロトコルの問題点

高品質なビデオデータ通信を実現するには、送信端から、通信経路上のルータを経由して、受信端に至る論理的なコネクション全体のQoS保証が必要である。また、このQoS保証は、リアルタイム・トラフィックのみでなく、FTP等の予測不能な非リアルタイム・トラフィックが混在する環境下において実現できなければならない。

さて、LANが相互に接続されたインターネット環境において、QoS保証を実現するには、まず、LAN内でのQoS保証を実現することが必須である。例えば、現在、広く普及している10Mbpsや100MbpsのEthernetを利用したLANの場合、QoS保証の実現には、物理セグメント内での帯域保証や遅延時間保証が課題となる。

Ethernetは、物理的な帯域幅だけを考慮すれば、圧縮ビデオデータ転送に十分な能力を有している。しかしながら、Ethernetは、伝送媒体へのアクセス競合をCSMA/CD方式により解決している。このため、短時間の間に多数のノードからの送信が集中した場合、トラフィックの上昇により、送信衝突の確率が上昇し、QoS保証が不能になる。

予測不能な非リアルタイム・トラフィックが混在するEthernet上において、高品質なリアルタイム通信を実現するには、物理的な伝送媒体を共有する全てのノードが発生させるトラフィックを抑制し、物理的な伝送媒体上の帯域を確保する手段が必要となる。

3.2. 総トラフィック量とサービス品質の関係

ここでは、Ethernetにおけるトラフィック量とサービス品質の関係を定量的に評価する。尚、以下では、共有ハブでノード間を接続した場合を含め、ひとつの物理セグメントを複数のノードが共有する場合を考える。

まず、QoSを保証するためには、データ転送の許容遅延時間（D）と同程度の時間間隔（T）毎にトラフィック量を制御しなければならない。例えば、許容できる遅延時間Dを10ミリ秒とした場合、時間間隔Tが500ミリ秒であると、500ミリ秒間隔で観測したトラフィック量が十分に小さくても、高トラフィック状態が10ミリ

秒を超えて継続する可能性があり、これは遅延時間保証の障害となる。

さて、物理セグメント内の総トラフィック量の制御を、時間間隔T毎の各ノードからの送信量の増減によって実現する場合、この時間間隔Tは、データ送信起動から次のデータ送信起動までの時間間隔（送信起動周期）に等価である。一方、データ送信時に衝突が発生する場合を考慮すると、この送信起動周期は、Ethernetのハードウェア再送機構が有効に機能できる程度に長くなければならない（注1）。これらの制約を考慮し、以下の議論では、10Mbps Ethernetの場合の送信起動周期40ミリ秒、100Mbps Ethernetの場合の送信起動周期5～10ミリ秒を仮定する。

図2に、10Mbps Ethernetにおける送信衝突による遅延時間と総送信要求量の関係を、計算機シミュレーションによって定量評価した結果を示す(各ノードからの送信要求量の総和とトラフィック量は、厳密には一致しないが、ここでは両者の違いは無視する)。図2の横軸は40ミリ秒毎に観測した総送信要求量、縦軸は最大送信遅延時間を示している。この評価では、1つの物理セグメント上に8台の送信ノードを接続した環境をシミュレートしている。デバイスドライバが発行する1回の送信起動によって、複数のパケットを連続送信可能なハードウェアを仮定し、送信起動間隔毎に、各ノードから送信されるパケット数を増減して、伝送路上のトラフィック量を制御している。

図2の結果から、総送信要求量がネットワークの物理的帯域幅の50%以下であれば、送信衝突はほとんど発生せず、遅延時間も無視できる程度に小さいことが判る。また、総送信要求量が70%程度までは、送信遅延時間を数ミリ秒以下に保つことができる。しかし、総送信要求量が75%を超えると、送信衝突の多発によって、遅延時間が増加する。総送信要求量が75%程度を超えると、各ノードでの再送回数の増加により、ネットワークのビジー率が大幅に増加し、これがさらにパケットの送信衝突を誘発する正帰還現象を発生する。その結果、衝突確率の増加により、再送の連続失敗回数が増加し、送信遅延時間が数百ミリ秒となる。さらに総送信要求量が増加すると、16回の再送上限回数を超過し、パケットの送信が中止される。ハードウェアによる再送が失敗し、TCP等のソフトウェアによる再送が動作した場合、対話的なビデオ通信アプリケーションが要求する遅延時間の保証はできない。

注1：Ethernetの物理層は、パケット送信の衝突検出機能と自動再送機能を有する。この自動再送機能は、衝突が発生した場合、ランダムな待ち時間の後、再送を試みる。再送時に再び衝突を検出すると、最大16回までの再送が試みられる。

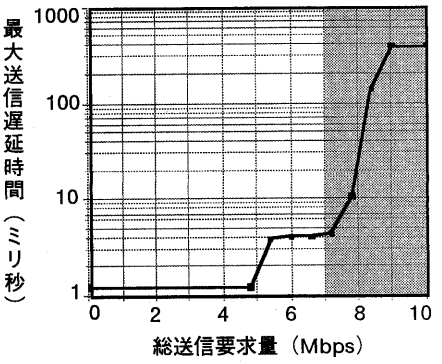

図2. Ethernetのトラフィック - 遅延特性

これらの結果から、Ethernetへの送信では、トラフィック量が70%程度までであれば、物理層の再送機能によって、数ミリ秒以下の遅延時間を高い確率で保証できることがわかる。

3.3. 総トラフィック制御プロトコルとアイソクロナス送信モード

前節に述べたEthernetの問題は、ひとつの物理セグメント上のトラフィック量が一定値を超えないように制御することで解決可能である。本節では、Ethernet上での遅延時間保証を実現する総トラフィック制御プロトコル（TTCP）とアイソクロナス送信モード（ITM）について述べる。

TTCP/ITM方式は、リアルタイム通信の遅延時間を保証するために、リアルタイム通信と非リアルタイム通信の両方のトラフィック量の合計を設定値以下に抑え、Ethernet上での高品質リアルタイム通信を可能とする。この方式の原理は、各ノードのネットワーク層（IP層）とデバイスドライバ層の間に、送信データ量を抑制するフィルタを設けることである。このフィルタは、帯域予約を行ったリアルタイム・パケットは全て通過させ、それ以外の非リアルタイム・パケットは、送信ノード自身が接続された物理セグメントを飽和させない様に、単位時間あたりの送信データ量を抑制する。即ち、物理セグメント内のリアルタイム・パケットと非リアルタイム・パケットのトラフィック量の総和が、許容トラフィック量を超えない様に、送信データ量をこのセグメント上の全ノード間で調停する。以下では、この具体的な実装方式を説明する。

3.3.1. 帯域管理サーバとTTCP

図3に示す様に、物理セグメント上にひとつの帯域管理サーバを設ける。この帯域管理サーバは、物理セグメント上の各ノードから発信された帯域割り当て要求を受信し、各ノードに使用帯域幅を割り当てる。各ノードは、リアルタイム通信、または、非リアルタイム通信を

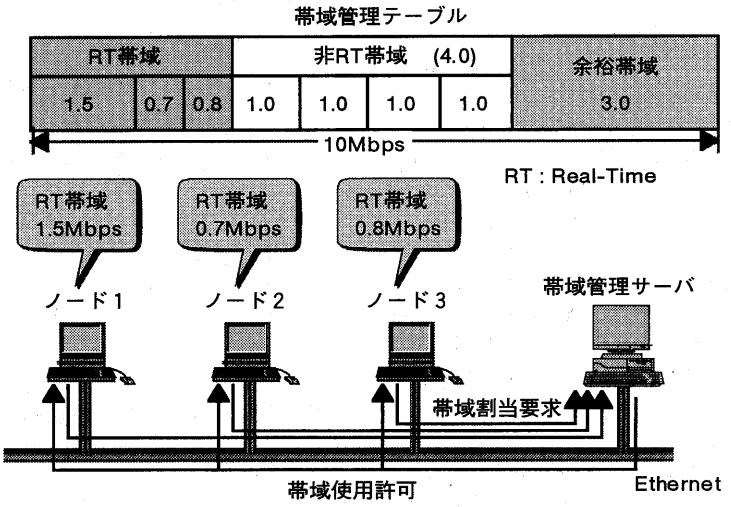

図3. TTCP（総トラフィック制御プロトコル）の処理概要

開始する前に、帯域割り当て要求を帯域管理サーバに送信する。TTCPでは、各ノードは、通信ポート単位に帯域の割り当てを要求する。各ノードは、リアルタイム通信を開始する場合、必要な帯域幅を指定した帯域割り当て要求を送る。また、非リアルタイム通信を開始する場合、帯域幅を指定しない帯域割り当て要求を送る。

帯域管理サーバは、リアルタイム通信の帯域割り当て要求を優先して帯域を割り当てる。リアルタイム通信の帯域割り当てが完了すると、過トラフィック状態の発生を避けるために、物理的な上限帯域幅からリアルタイム通信用帯域幅と余裕帯域幅を減じる。さらに、残った帯域幅を非リアルタイム通信用として要求ノード数で等分割する。我々の実験システムでは、余裕帯域幅を20～30%、リアルタイム通信用帯域幅の上限を50%に設定し、最低でも20～30%の非リアルタイム通信用帯域幅が確保できる様に設定している。

各ノードへの帯域割り当て処理が完了すると、帯域管理サーバは各要求ノードへ帯域割り当て情報を返送する。TTCPでは、原則的に、リアルタイム通信用帯域は、アプリケーションからの明示的な要求によって解放されるまで帯域割り当てが継続される。一方、非リアルタイム通信用帯域は、一定の時間、データ転送が行われないと自動的に解放される。但し、アプリケーションが異常終了した場合には、それが確保していた帯域は自動的に解放される。また、ノードが予想外の障害によって停止した場合、このノードに割り当てられていた全帯域は自動的に解放される。各ノードと帯域管理サーバは、障害発生等による予約帯域の解放漏れを防ぐために、割り当て帯域の使用継続要求メッセージや継続確認メッセージを定期的（約10秒間隔）に交換する。

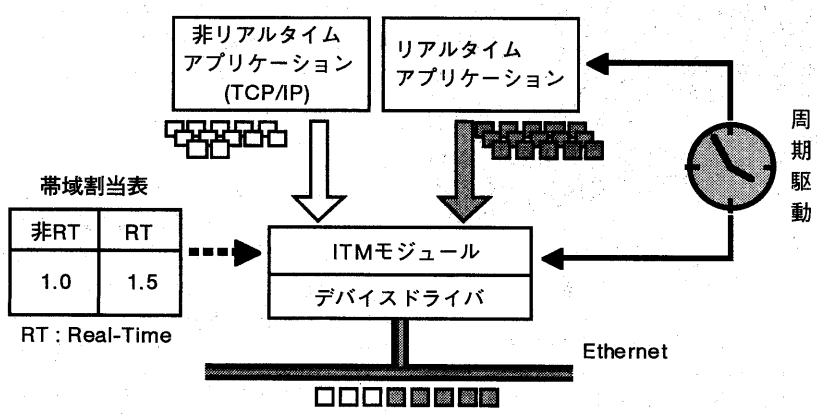

図4. ITMモジュールの処理概要

3.3.2. アイソクロナス送信モードと送信抑制

アイソクロナス送信モードは、前節に述べた帯域管理サーバからの帯域割り当て情報に従い、自ノードからの送信データ量を抑制し、物理セグメント内の総トラフィック量を一定値以下に抑制する。具体的には、図4に示す様に、各ノードに組み込まれたITMモジュールが、アプリケーションから発行された送信要求の実行をデバイスドライバの直前で一旦保留し、数十ミリ秒の短い時間間隔で周期的に送信ハードウェアを起動し、一周期内の自ノードからの送信データ量を調整し、伝送路上の総トラフィック量を一定の上限値以下に抑制する。

ITMモジュールによる送信起動周期毎の送信起動は、アイソクロナス・スケジューリングによって実現できる。送信起動周期は、同一物理セグメント上の全ノードで同一でなければならないが、一周期の開始時刻はノード間で同期させる必要はない（注2）。ノード間で送信タイミングを調整しないので、送信起動周期よりも短い時間内では、一時的に送信衝突が発生する。しかし、Ethernet物理層の再送機構によって、各ノードはランダム時間後に再送を試みる。送信起動周期内の平均的なトラフィック量が一定の上限値を超えない様に制御しているので、この再送時に繰り返し送信衝突が発生する確率は十分に低く、各ノードは高い確率で送信起動周期以内に、1周期分の全パケットの送信を完了できる。即ち、リアルタイム通信については、1周期毎に、予約帯域幅を送信周期で分割した量のパケットが伝送路へ送出される。同様に、非リアルタイム通信については、1周期毎に、帯域管理サーバが割り当てた帯域幅を送信周期で分割した量のパケットが伝送路へ送出される。

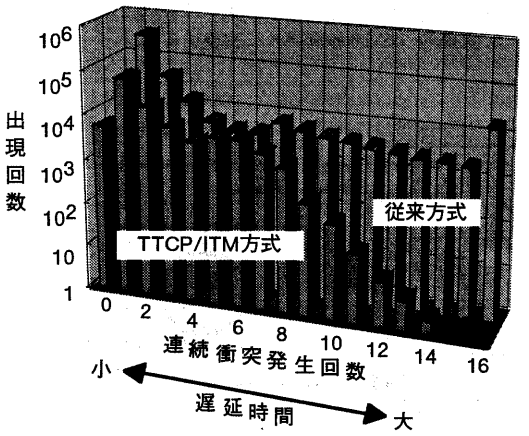

図5. トラフィック抑制の実測結果

注2：ノード間で送信タイミングが集中すると送信衝突率が上昇し、遅延時間が多少増加する。各ノードの送信時刻は、ランダムに分散している方が好ましい。

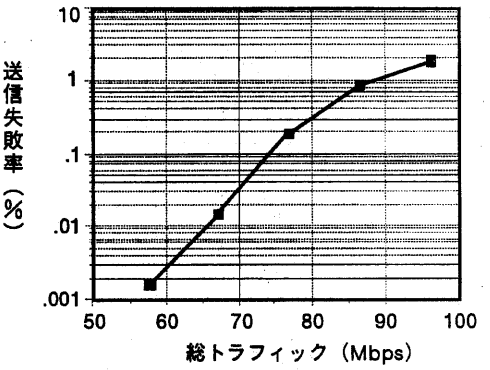

図6. 総トラフィック - 送信失敗率の実測結果

4．性能評価

TTCP/ITMを実装した8ノード（内2ノードはFreeBSD搭載、6ノードはTactix搭載）を100Mbps Ethernet共有型ハブで接続し、リアルタイム・トラフィックと非リアルタイム・トラフィックを混在させて、連続送信衝突回数の分布を測定した。図5の前側のグラフは、総トラフィック量を48%に抑制した場合、図5の奥側のグラフは、総トラフィック量が96%に達した場合の連続送信衝突回数の分布を示す。総トラフィック量を抑制することにより、ハードウェア機構による再送が有効に機能していることが確認できる。

図6のグラフは、総トラフィック量を変化させた場合の送信失敗率（16回の再送が全て失敗する確率）を示す。このグラフから、総トラフィック量を75%以下にすれば、99.9%以上のパケットを十分小さな遅延時間内に転送できることがわかる。

5．おわりに

本稿では、アイソクロナス・スケジューリングとこれを応用したEthernet上でのリアルタイム通信方式について述べた。ここに述べたQoS保証技術とその評価実験結果は、これまで困難であると考えられてきたEthernet上での高品質なビデオ・データ転送を実現できる可能性を示唆している。

参考文献

[1] 岩嵜他，「連続メディア処理向きマイクロカーネルHiTactixの設計と評価」，情報処理学会，96年コンピュータシステム・シンポジウム論文集，Nov. 96

[2] 竹内他，「アイソクロナス・スケジューラの設計と性能評価」，情報処理学会，システムソフトウェアとOS研究会，マルチメディア通信と分散処理研究会，Feb. 97

[3] M. Iwasaki, et. al, "A Micro-kernel for Isochronous Video-Data Transfer," WWCA97, Mar. 97