情報処理学会論文誌 Vol.56 No.9 1905–1915 (Sep. 2015)

EBRRパケットスケジューリングアルゴリズムの拡張提案

松田 哲史[1,a)]

受付日 2014年8月2日, 採録日 2015年6月5日

概要：Lenzini らが提案した Eligibility Based Round Robin（EBRR）パケットスケジューリングアルゴリズムは，Surplus Round Robin（SRR）を設定パラメータと最大パケットサイズの大小関係によらず計算量が O(1) となるように拡張したものと考えることができる．本論文では，計算量が O(1) の性質を保ちながら，1 スケジューリングラウンド内に受信する複数パケットの中で閾値よりも小さいサイズのパケットを先に送信することで，小サイズパケットの遅延時間を減らすための EBRR 拡張方式を提案する．そして，提案方式の有効性をシミュレーションで評価した結果と，パケット遅延時間上限と公平性の解析結果を示す．提案するパケットスケジューリングアルゴリズムをホームゲートウェイ等の宅内通信機器に適用することで，VoIP 等の遅延時間の影響を受けやすい通信の遅延時間を減らすことが可能となる．

キーワード：パケットスケジューリング，EBRR

A Proposal of an Extension of EBRR Packet Scheduling Algorithm

Tetsushi Matsuda[1,a)]

Received: August 2, 2014, Accepted: June 5, 2015

Abstract: Eligibility Based Round Robin (EBRR) scheduler proposed by Lenzini et al. is a packet scheduling algorithm which can be considered as an extension of Surplus Round Robin (SRR), of which computational complexity is O(1) even when configuration parameter is less than the maximum packet size. In this paper, we propose an extension of EBRR which makes it possible to transmit packets smaller than a threshold before packets larger than the threshold among packets received in one scheduling round. Evaluation result by simulation and analysis of latency bound and fairness are shown. The proposed algorithm can be used in customer premises equipments such as Home Gateway so that the latency of traffic such as VoIP which are sensitive to latency can be reduced.

Keywords: packet scheduling, EBRR

1. はじめに

Deficit Round Robin（DRR）[1], [2]，Surplus Round Robin（SRR）[3], [4], Nested DRR [5], Elastic Round Robin（ERR）[6], LBFS-DRR [7], Eligibility Based Round Robin（EBRR）[8] 等，計算量が O(1) のパケットスケジューリングアルゴリズムが多数提案されている．Lenzini らが提案した EBRR は，Surplus Round Robin（SRR）を設定パラメータと最大パケットサイズの大小関係によらず計算量が O(1) となるように拡張したものと考えることができる．EBRR では，同一スケジューリングラウンドにスケジュールされたパケットの中で，小さいサイズのパケットを大きいサイズのパケットより先に送信する仕組みは提供されていない．このため，大きいサイズのパケットが小さいサイズのパケットより先に送信されると，大きいサイズのパケットの出力インタフェースでの伝送が完了するまで小さいサイズのパケット送信を開始できないため，小さいサイズのパケットにより大きな遅延時間が生じる可能性がある．遅延時間が増加すると通話品質が低下する VoIP パケットや，遅延時間増による往復遅延時間増がスループット低下につながる TCP ACK といったパケットが小さいサイズで

[1] 三菱電機株式会社情報技術総合研究所
Information Technology R&D Center, Mitsubishi Electric Corporation, Kamakura, Kanagawa 247–8501, Japan
[a)] Matsuda.Tetsushi@dh.MitsubishiElectric.co.jp

あることを考慮すると，小さいサイズのパケットを大きいサイズのパケットより先に送信することに，通信品質上のメリットがあると考えられる．VoIP 等のパケット送信を優先する方法として，パケット転送優先度を示す IP ヘッダ内フィールドの値に基づく Priority Queueing（PQ）によるパケットスケジューリングを用いる方法が知られている．パケット転送優先度と PQ を用いる方法では，各パケット転送優先度に対する帯域制御機能がないため，高パケット転送優先度の通信が全帯域を使い，低パケット転送優先度の通信ができなくなる可能性がある．しかし，システムによっては，低パケット転送優先度の通信にも最低保証帯域を確保し，すべての優先度の通信を同時に行うことが必要な場合がある．このため，帯域制御と，小さいサイズのパケットを先に送信することを両立できる方法が必要となる．

本論文では，計算量，遅延時間，公平性に優れ，最低保証帯域を実現可能な EBRR をもとに，計算量が O(1) のまま，1 スケジューリングラウンド内に受信する複数パケットの中で閾値よりも小さいサイズのパケットを先に送信することで，小サイズパケットの遅延時間を減らすための拡張方式を提案する．提案方式は文献 [10], [11] で提案した方式に拡張を行ったものであり，加えて，提案方式でのパケット遅延時間最大値と公平性の解析結果を示す．提案方式を，VoIP，Web アクセス，P2P 等様々なアプリケーションのトラヒックを同時に転送するホームゲートウェイ等の宅内通信機器に適用することで，VoIP や TCP ACK のように小サイズで遅延時間の影響を受けやすいパケットの遅延時間を減らすことが可能となる．

本論文の構成は以下のとおりである．2 章で，拡張対象となる EBRR と，提案方式が解決しようとする問題点を説明する．3 章で 3 つの拡張項目で構成される提案方式を説明する．4 章で提案方式の効果を，シミュレーションにより検証する．5 章で遅延時間上限と公平性の解析結果を示す．6 章で既存研究と提案方式の比較を行い，7 章でまとめを述べる．

2. EBRR と解決対象とする問題の説明

2.1 EBRR の説明 [8]

拡張対象となる EBRR の処理内容を説明する．EBRR では，パケットスケジューリングをパケットヘッダ内容等で識別される flow の単位で行う．スケジューリング対象の flow 数を N とし，i 番目の flow を $\mathrm{flow_i}$ と表記する．$\mathrm{flow_i}$ のパケットの最大サイズを $\mathrm{L_{max,i}}$ とする．i 番目 $(\mathrm{i} = 1, .., \mathrm{N})$ の $\mathrm{flow_i}$ ごとに，以下のデータを管理する．

(a) $\mathrm{flow_i}$ に属する送信待ちパケットを保持するキュー Queue[i].

(b) パケット送信タイミング判断に使用する Credit Counter $\mathrm{C_i}$.

(c) quantum 値 $\mathrm{Q_i}$．quantum の値は，flow に指定される送信レート $\mathrm{r_i}$ に対して，任意の $\mathrm{flow_i}$ と $\mathrm{flow_j}$ $(\mathrm{i} \neq \mathrm{j})$ の間で $\mathrm{Q_i} : \mathrm{Q_j} = \mathrm{r_i} : \mathrm{r_j}$ となるように決める．$\mathrm{flow_i}$ に属する送信待ちパケットが存在する場合，スケジューリングラウンドごとに $\mathrm{Q_i}$ を $\mathrm{C_i}$ に加算する．本論文では，$\mathrm{Q_i}$ の値が $\mathrm{L_{max,i}}$ より小さい場合を想定する．

(d) Queue[i] が空の状態で $\mathrm{flow_i}$ に属するパケットを受信したときに，そのパケットを送信可能となるスケジューリングラウンド（$0, 1, \ldots$ の非負整数値）を示す $\mathrm{rc_i}$．初期値は 0.

また，flow に依存しないデータとして，以下のデータを管理する．

(e) 現在のスケジューリングラウンド（$0, 1, \ldots$ の非負整数値）を示すカウンタ RC.

(f) 全フローの quantum $\mathrm{Q_i}$ の中の最小値 $\mathrm{Q_{min}}$.

(g) Queue[i] のリストを保持する配列 ActiveList[]. ActiveList[] のサイズ q = ceil($\mathrm{L_{max,i}}$ の最大値/$\mathrm{Q_{min}}$) とする．(ceil(x) は x 以上の最小の整数を返す関数) ActiveList[] のインデックスはスケジューリングラウンドに対応付けられ，スケジューリングラウンド rc にスケジューリング対象となる Queue[i] は，ActiveList[rc mod q] が表すリストの要素である（mod は modulo を表す）.

上記 (a) から (g) のデータを用いて，以下のアルゴリズムでスケジューリング処理を行う．i) は，出力インタフェースで 1 パケットの送信完了時に呼び出される．

i) ActiveList[RC mod q] が空ならば，RC を 1 増やして次のスケジューリングラウンドを開始し，i) を実行．空でなければ，最初の要素 (Queue[i]) を ActiveList[RC mod q] からデキューする．そして Queue[i] の先頭からパケットを 1 つデキューして送信し，$\mathrm{C_i}$ をパケットサイズ分減算する．上記処理の後に ii) の処理に移る．

ii) $\mathrm{C_i} > 0$ かつ Queue[i] が空でなければ，Queue[i] を ActiveList[RC mod q] の最後に追加する．
$\mathrm{C_i} \leqq 0$ ならば，$\mathrm{Q_i}$ をスケジューリングラウンドごとに加算することで $\mathrm{C_i} > 0$ となるスケジューリングラウンド $\mathrm{rc_i}$ を求める．$\mathrm{rc_i} = \mathrm{RC} + \mathrm{floor}(-\mathrm{C_i}/\mathrm{Q_i}) + 1$ となる（floor(x) は x 以下の最大整数を返す関数）．この場合，Queue[i] が空でなければ Queue[i] を ActiveList[$\mathrm{rc_i}$ mod q] の最後に追加し，$\mathrm{C_i}$ を $\mathrm{rc_i}$ での Credit Counter 値に設定する．Queue[i] が空ならば，$\mathrm{flow_i}$ に属する新規受信パケットは $\mathrm{rc_i}$ 以降のスケジューリングラウンドで送信可能となることを記録し，$\mathrm{C_i} = \mathrm{Q_i}$ とする．上記処理の後に i) に移る．

iii) $\mathrm{flow_i}$ に属するパケットを受信したら Queue[i] にエンキューする．エンキューする前に Queue[i] が空であった場合，Queue[i] を ActiveList[max($\mathrm{rc_i}$, RC) mod q] の最後に追加する（max(x, y) は x と y の最大値を返す関数）．パケット送信中でなければ i) に移る．

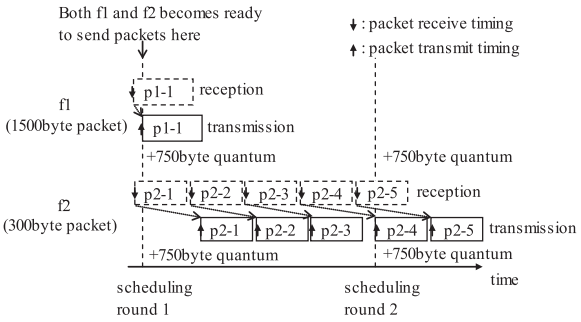

図 1 EBRR のパケットスケジューリング例
Fig. 1 Example packet scheduling by EBRR.

2.2 解決対象とする問題の説明

本論文の提案方式が解決対象とする，EBRR の問題点を説明する．f1 と f2 の 2 つの flow が定義され，f1 の quantum 値 Q_1 と f2 の quantum 値 Q_2 のいずれもが 750 Byte の場合を考える．f1 と f2 のいずれもがスケジューリングラウンド 1 でパケット送信可能となると仮定する．f1 に属する 1500 Byte のパケット p1-1 と，f2 に属する 300 Byte のパケット p2-1 が，この順序でスケジューリングラウンド 1 の開始直前に受信され，スケジューリングラウンド 1 の間に，f2 に属する p2-2 から p2-5 の 4 つの 300 Byte パケットが続けて受信される場合を考える（**図 1**）．

2.1 節に示した EBRR アルゴリズムに従うと，スケジューリングラウンド 1 に対応する ActiveList[] 要素に Queue[f1], Queue[f2] の順でリストされ，p1-1，p2-1，p2-2，p2-3 の順にスケジューリングラウンド 1 でパケットが送信される．そしてスケジューリングラウンド 2 に対応する ActiveList[] 要素に Queue[f2] がリストされ，p2-4，p2-5 の順にスケジューリングラウンド 2 でパケットが送信される（図 1）．

上記のパケット送信順では，p2-1 の出力インタフェース上の送信は，p1-1 の出力インタフェース上の送信が完了するまで待つ必要があり，p2-1 がより長く遅延することとなる．p1-1 が動画配信のようなアプリケーションのパケットで，p2-1 が VoIP のパケットである場合，p2-1 のアプリケーションは p1-1 のアプリケーションより遅延時間の影響を受けやすいので，p2-1 の遅延時間を短くするために p2-1 を先に送信し p1-1 を後で送信することが望ましいと考えられる．一般的に動画配信パケットはサイズが大きく，VoIP パケットはサイズが小さい．また，動画配信パケットと VoIP パケットはともにパケットロス発生が望ましくない．このため，パケット転送優先度が同じ場合を想定すると，パケット転送優先度を示す IP ヘッダフィールドである DSCP 値 [9] が同一の flow 間で，小さいサイズのパケットを大きいサイズのパケットよりも先に送信することが有効な場合があると考えられる．

上記の EBRR のパケットスケジューリング動作は EBRR の基である SRR に由来するものであり，図 1 の例で p2-1 の遅延時間を減らすために，p2-1 が p1-1 よりも先に送信されるようにするには EBRR アルゴリズムの拡張が必要となる．3 章に拡張内容を示す．

3. 提案方式の説明

3.1 スケジューリングアルゴリズムの説明

本節では提案する EBRR 拡張方式を説明する．提案する拡張は 3 つの拡張項目で構成される．文献 [10], [11] で提案した方式に対して，第 2 の拡張項目に THRESH による判定条件を追加することで，遅延時間をより削減可能とした．

第 1 の拡張項目は ActiveList[] を 2 つの配列に分割することである．2 つの配列は，閾値 THRESH 未満のサイズのパケットが先頭にある Queue[i] を要素とするリストが配列要素となる ActiveListS[] と，閾値 THRESH 以上のサイズのパケットが先頭にある Queue[i] を要素とするリストが配列要素となる ActiveListL[] である．EBRR アルゴリズムで ActiveList[] に Queue[i] を追加する処理を，Queue[i] の先頭パケットサイズにより ActiveListS[] と ActiveListL[] のいずれかに追加する処理に変更する．また，EBRR アルゴリズムで ActiveList[] から Queue[i] を取り出す処理を，ActiveListS[] に要素が存在する間は ActiveListS[] から Queue[i] を取り出し，ActiveListS[] が空の場合に ActiveListL[] から Queue[i] を取り出す処理に変更する．Queue[i] をいずれに追加するかの判断に必要な処理は，Queue[i] の先頭パケットのサイズと THRESH との比較だけなので，本拡張を行ってもアルゴリズムの計算量が O(1) であることは変わらない．本拡張により，同一スケジューリングラウンドにスケジュールされるパケットの中で，サイズが THRESH より小さいパケットが，サイズが THRESH 以上のパケットより先に送信されることとなる．

第 2 の拡張項目のために，もう 1 つの閾値 TH を定義する．TH は 0 以下の値をとる．そして，パケット送信可能と判断する条件を，EBRR での “$C_i > 0$” から，“パケットサイズが THRESH 未満の場合は (C_i – パケットサイズ) > TH，THRESH 以上の場合は (C_i – パケットサイズ) > 0” に変更する．条件が満たされない場合は，1 スケジューリングラウンドごとに Q_i を加算することで条件が満たされるスケジューリングラウンドに，パケット送信をスケジュールする．つまり，より大きいサイズのパケットは C_i の値がより大きくなるスケジューリングラウンドまで送信を待たされるように変更する．パケットスケジューリングアルゴリズムの中で Queue[i] の先頭パケットをデキューする場合は，少なくとも 1 つのパケットが送信され，計算量が O(1) であることを保証するために，パケット受信時とパケット送信後のスケジューリング処理における，パケットを送信可能なスケジューリングラウンド計算処理を変更することが必要となる．本拡張により，同一スケジューリングラウンドで受信したパケットの中で，flow の Credit Counter

値と TH の値で決まる閾値より大きなパケットは後のスケジューリングラウンドにスケジュールされることとなる．

第 3 の拡張項目のために，各 $flow_i$ ごとの最大バーストサイズを指定する設定パラメータ max_burst_i を導入する．$flow_i$ の C_i を，EBRR アルゴリズムの iii) において rc_i から RC の間スケジューリングラウンドごとに Q_i ずつ増加させ最大 max_burst_i まで増加可能とする．これにより，$flow_i$ へのパケット到着を待つ間に Credit Counter の値を max_burst_i まで増加可能とする．本拡張により，$flow_i$ にパケットが到着するのを待つ間に，他フローでパケット送信が行われたためにスケジューリングラウンドが進んだ場合に，$flow_i$ の Credit Counter の値を増やすことで，$flow_i$ に到着したパケットの送信待ち時間を減らす効果が得られる．

上記 3 つの拡張項目を反映した拡張スケジューリングアルゴリズムを以下に示す．i′) は，出力インタフェースで 1 パケットの送信完了時に呼び出される．

i′) ActiveListS[RC mod q] と ActiveListL[RC mod q] の両方が空ならば，RC を 1 増やして次のスケジューリングラウンドを開始し，i′) を実行する．
ActiveListS[RC mod q] が空でなければ，ActiveListS[RC mod q] の最初の要素（Queue[i]）を取り出す．
ActiveListS[RC mod q] が空で，ActiveListL[RC mod q] が空でなければ，ActiveListL[RC mod q] の最初の要素（Queue[i]）を取り出す．
取り出した Queue[i] の先頭パケットをデキューして送信し，C_i をパケットサイズ分減算し，ii′) の処理に移る．

ii′) Queue[i] が空でなく，条件 1 {Queue[i] の先頭パケットサイズが THRESH 未満で (C_i − Queue[i] の先頭パケットサイズ) > TH か，Queue[i] の先頭パケットサイズが THRESH 以上で (C_i − Queue[i] の先頭パケットサイズ) > 0} が満たされるならば，Queue[i] の先頭パケットのサイズにより Queue[i] を ActiveListS[RC mod q] か ActiveListL[RC mod q] の最後に追加する（Queue[i] の先頭パケットのサイズが THRESH 未満ならば ActiveListS[RC mod q] に追加し，そうでなければ ActiveListL[RC mod q] に追加する）．
条件 1 が満たされない場合，スケジューリングラウンドごとに quantum 値 Q_i を加算することで条件 1 が満たされるスケジューリングラウンド rc_i を求める（i.e., Queue[i] の先頭パケットサイズが THRESH 未満ならば $rc_i = RC + floor((Queue[i] の先頭パケットサイズ + TH − C_i)/Q_i) + 1$，そうでなければ $rc_i = RC + floor((Queue[i] の先頭パケットサイズ − C_i)/Q_i) + 1$）．そして，Queue[i] の先頭パケットサイズに従って ActiveListS[rc_i mod q] か ActiveListL[rc_i mod q] の最後に Queue[i] を追加し，C_i を rc_i における C_i の値に設定する．Queue[i] が空の場合，スケジューリングラウンドごとに quantum 値 Q_i を加算することで $C_i > TH$ となるスケジューリングラウンド rc_i を求める（$C_i \leqq TH$ ならば $rc_i = RC + floor((TH - C_i)/Q_i) + 1$，$C_i > TH$ ならば $rc_i = RC$）．そして C_i を rc_i での値に設定する．
上記処理の後に i′) の処理に移る．

iii′) $flow_i$ に属するパケットを受信したら Queue[i] にエンキューする．
エンキューする前は Queue[i] が空であった場合，rc を Queue[i] の先頭パケットを送信可能なスケジューリングラウンドとして，Queue[i] を ActiveListS[rc mod q] か ActiveListL[rc mod q] の最後に，Queue[i] の先頭パケットサイズによって追加する．rc は次のように求める．$RC > rc_i$ ならば $C_i = \min(C_i + Q_i * (RC - rc_i), max\_burst_i)$ とする．受信パケットサイズが THRESH 未満ならば，(C_i − 受信パケットサイズ) > TH なら $rc = \max(rc_i, RC)$，そうでなければ $rc = \max(rc_i, RC) + floor((受信パケットサイズ + TH - C_i)/Q_i) + 1$ とする．受信パケットサイズが THRESH 以上ならば，(C_i − 受信パケットサイズ) > 0 なら $rc = \max(rc_i, RC)$，そうでなければ $rc = \max(rc_i, RC) + floor((受信パケットサイズ - C_i)/Q_i) + 1$ とする．パケット送信中でなければ i′) に移る（min(x, y) は x と y の最小値を返す関数）．

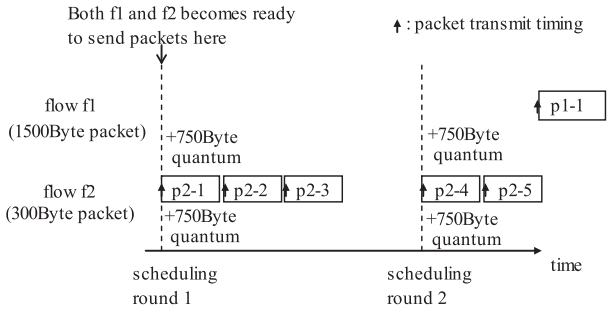

図 2 提案方式のパケット送信シーケンス例

Fig. 2 Example packet transmit sequence by proposed algorithm.

3.2 パラメータ TH の効果

図 1 のパケット受信シーケンスに対する提案方式でのスケジューリング例で，パラメータ TH の効果を説明する．THRESH の値を 301 Byte，TH の値を −300 Byte に設定した場合を考える（図 **2**）．

p1-1 受信時に，スケジューリングラウンド 1 開始時点での C_1 の値は 750 であり，パケットサイズ > THRESH で (C_1 − パケットサイズ) = (750 − 1500) = −750 < 0 となるため，p1-1 はスケジューリングラウンド 2 で送信することになる．P2-1 受信時に，スケジューリングラウンド 1 開始時点での C_2

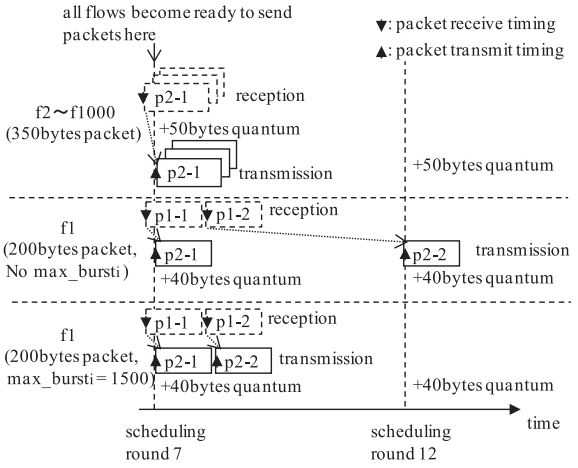

図 3 max_burst_i がある場合とない場合の比較

Fig. 3 Comparison between the case with max_burst_i and the case without it.

の値は 750 であり，パケットサイズ < THRESH であるため，スケジューリングラウンド 1 で p2-1, p2-2, p2-3 を送信すると (C_2−パケットサイズ) = (750−300∗3) = −150 > TH だが，p2-1, p2-2, p2-3, p2-4 を送信すると，(C_2−パケットサイズ) = (750−300∗4) = −450 < TH となるので，p2-4, p2-5 はスケジューリングラウンド 2 で送信することとなる．スケジューリングラウンド 2 では，300 < THRESH < 1,500 なので，Queue[f1] が ActiveListL[] の要素，Queue[f2] が ActiveListS[] の要素となり，p2-4，p2-5，p1-1 の順で送信される．以上より，THRESH と TH の効果で，パケットサイズが小さい p2-1 から p2-5 が p1-1 よりも先に送信されるスケジューリングが行われることが分かる．

3.3 パラメータ max_burst_i の効果

quantum 値が 40 byte で 200 byte のパケットを送信する f1 の 1 個の flow と，quantum 値が 50 byte で 350 byte のパケットを送信する f2 から f1000 までの 999 個の flow が存在する場合のスケジューリング例で，パラメータ max_burst_i の効用を説明する．THRESH の値を 201 Byte，TH の値を −200 Byte，max_burst_i の値を 1,500 Byte に設定する．スケジューリングラウンド 7 開始時点で f2 から f1000 で 350 byte パケットを 1 つずつ受信済みで，パケット送信を開始しているとする（図 **3**）．

この状態で，f1 で 200 byte パケットを 2 つ受信した場合を考える．max_burst_i がない場合，1 つ目のパケット受信時の Credit Counter C_1 は 40 となり，1 つ目のパケットはスケジューリングラウンド 7 で送信可能（送信後の $C_1 = -160$）となるが，2 つ目のパケットは C_1 が 40 に回復するスケジューリングラウンド 12 まで送信できないため，f2 から f1000 のパケット送信完了後まで 2 つ目のパケット送信は待たされる．max_burst_i がある場合，1 つ目のパケット受信時の Credit Counter C_1 は 7 * 40 = 280 となり，2 つ目のパケットもスケジューリングラウンド 7 で送信可能（送信後の $C_1 = -120$）となり，f2 から f1000 のパケット送信完了を待たずに 2 つ目のパケット送信が可能となる．quantum 値が 40 byte で 200 byte のパケットを送信する f1 同様の flow が複数同時に存在する場合，f1 についての上記説明がそれらの flow 各々について同様に成り立つ．よって，それら複数の flow のパケットが f2 から f1000 より先に送信されることとなる．

3.4 スケジューリングアルゴリズムのためのパラメータ取得方法について

提案方式は，ホームゲートウェイ等の宅内通信機器を適用機器の例として想定している．ネットワークの出入口に配置されるホームゲートウェイのような通信機器でフローごとの帯域制御等の高機能なパケットスケジューリングを行い，ネットワーク内部のスイッチやルータでは，帯域制御されたトラフィックに十分な帯域が確保されているという前提で，フローごとの通信品質を保証するネットワークを構成することを想定している．このため，通信アプリケーションの性質を知ることができるエンドユーザが，フローごとの quantum 値を計算する元データとなる最低保証帯域値，TH，max_burst_i の値を計算したうえで，宅内のホームゲートウェイに設定することが考えられる．もしくは，通信に使用される TCP/UDP のポート番号やプロトコル種別をもとに，ホームゲートウェイが通信アプリケーションを特定し，各通信アプリケーションに対して事前に定義される特性情報をもとに，フローごとの quantum 値，TH，max_burst_i の値を自動的に計算し設定する方法が考えられる．

4. 提案方式の効果検証

本章では，提案方式の効果を 5 つのケースについてシミュレーションで検証した結果を述べる．すべてのケースで，4 つの 1 Gbps 入力ポート（port1–port4）からパケットを受信し，100 Mbps の 1 出力ポートから送信すると仮定した．出力ポートの送信レートを 100 Mbps に設定した理由は，現在の光アクセスサービスに用いられる GE-PON は最大 32 加入者で 1 Gbps の帯域を共有する方式であり [12]，複数加入者が同時に通信を行う場合に 1 加入者が利用可能な帯域が 1 Gbps から減少するためである．シミュレーションに用いた A) から E) の 5 ケースの内容を以下に示す．1,500 byte パケットの flow で P2P や Web アクセス等のデータ通信を模擬し，200 byte パケットの flow で VoIP 通信を模擬する．VoIP 通信を模擬するパケットを 200 byte に設定した理由は，VoIP 用音声コーデックとして用いられる ITU-T G.711 μ-law で 20 ms 間隔で音声パケットを送信する場合，IPv4 ヘッダ（20 byte），UDP ヘッダ（8 byte），RTP ヘッダ（12 byte），音声データ（160 byte）の合計が

200 byte となるためである．

A) 1,500 byte パケットを 3.003 Mbps で一定時間間隔で受信する 333 個の flow が port1，2，3 に存在し（各 port に 1 Gbps），合計 999 個の flow（flow1–flow999）が存在．flow1–flow999 の各 flow の最低保証帯域値は 100.02 kbps とする．200 byte パケットを 20 ms に 1 回受信（80 kbps）する 1 つの flow（flow1000）が port4 に存在する．flow1000 の最低保証帯域値は 80.016 kbps とする．シミュレーション対象時間は 20 秒間で，シミュレーション対象時間中 flow1–flow999 の各 flow に少なくとも 1 つの送信待ちパケットが存在する．

B) 1,500 byte パケットを 30.3 Mbps で一定時間間隔で受信する 33 個の flow が port1，2，3 に存在し（各 port に 1 Gbps），合計 99 個の flow（flow1–flow99）が存在．各 port 上では各 flow のパケットを巡回的に受信する（i.e., port1 では flow1, flow2, ..., flow33, flow1, ... の順で受信）．flow1–flow99 の各 flow の最低保証帯域値は 1.009 Mbps とする．200 byte パケットを 20 ms に 1 回受信（80 kbps）する 1 つの flow（flow100）が port4 に存在する．flow100 の最低保証帯域値は 80.74 kbps とする．シミュレーション時間は 20 秒間で，flow1–flow99 の各 flow に少なくとも 1 つの送信待ちパケットが存在する．

C) シミュレーション時間を 12 ms 単位の区間に分割する．1,500 byte パケットを 1 Mbps で受信する 33 個の flow が port1，2，3 に存在し（各 port に 33 Mbps），合計 99 個の flow（flow1–flow99）が存在．各 flow のパケットは各 12 ms 区間の中のランダムなタイミングで 1 回受信する．flow1–flow99 の各 flow の最低保証帯域値は 1.009 Mbps とする．200 byte パケットを 20 ms に 1 回受信（80 kbps）する 1 つの flow（flow100）が port4 に存在する．flow100 の最低保証帯域値は 80.74 kbps とする．シミュレーション時間は 20 秒間で，異なる乱数シードで 11 回シミュレーションを行った．

D) 1,500 byte パケットを 1 Mbps で受信する 33 個の flow が port1，2，3 に存在し（各 port に 33 Mbps），合計 99 個の flow（flow1–flow99）が存在．各 flow のパケットはシミュレーション時間間隔の中のランダムなタイミングで 1 Mbps 分のパケット数受信する．flow1–flow99 の各 flow の最低保証帯域値は 1.009 Mbps とする．200 byte パケットを 20 ms に 1 回受信（80 kbps）する 1 つの flow（flow100）が port4 に存在する．flow100 の最低保証帯域値は 80.74 kbps とする．シミュレーション時間は 20 秒間で，異なる乱数シードで 11 回シミュレーションを行った．

E) 200 byte パケット flow が複数存在する場合の評価のため，ケース D) に対して，200 byte パケットを 20 ms に 1 回受信（80 kbps）する flow を 3 つ（flow101–flow103）を port4 に追加．flow1–flow99 の各 flow の最低保証帯域値は 1.007 Mbps とする．flow100–flow103 の各 flow のパケット到着時刻は 5 ms 間隔でずらす．flow100–flow103 の最低保証帯域値は 80.55 kbps とする．シミュレーション時間は 20 秒間で，異なる乱数シードで 11 回シミュレーションを行った．

表 1 シミュレーションパラメータ
Table 1 Simulation parameters.

	$Q_{1\text{-}99/999}$	$Q_{100\text{-}103/1000}$	THRESH	TH	max_burst 1-99/999	max_burst 100-103/1000
A)	50	40	201	-200	3000	1500
B)	50	4	201	-200	3000	1500
C)	50	4	201	-200	3000	1500
D)	50	4	201	-200	3000	1500
E)	50	4	201	-200	3000	1500

表 2 最大と平均遅延時間のシミュレーション結果
Table 2 Maximum and average delay result of simulation.

	EBRR		Proposed Algorithm	
	maximum delay	average delay	maximum delay	average delay
A)	119.86ms	67.86ms	0.12ms	0.059ms
B)	11.87ms	5.95ms	0.12ms	0.059ms
C)	9.54ms	2.18ms	0.12ms	0.060ms
D)	9.15ms	1.76ms	0.12ms	0.059ms
E)	9.91ms	2.03ms	0.12ms	0.059ms

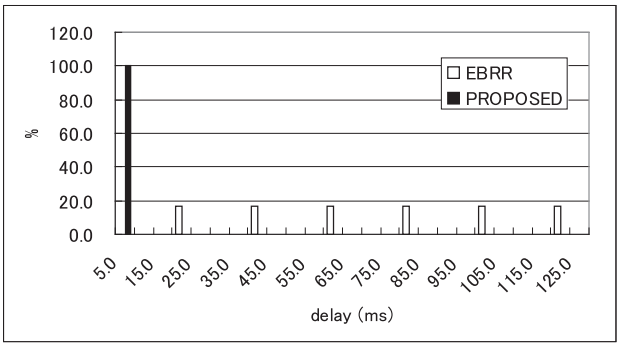

図 4 ケース A) での遅延時間分布（5 ms 間隔）
Fig. 4 Delay distribution of case A) (5 ms interval).

表 1 にシミュレーションに使用したパラメータ値を示す．同じ Q_i の値を EBRR のシミュレーションにも用いた．

Q_1 から Q_{1000} の値は，最低保証帯域値の比に基づいて決定している．遅延時間を減らす対象となる小サイズパケットが 200 byte であるので，THRESH の値を 201 byte に設定し，TH の値を −200 byte に設定した．max_burst_i の値は，遅延時間を減らす対象となる小サイズパケットを受信しない間に増加するスケジューリングラウンド数分 Credit Counter C_i の値が増加できる程度の大きさに設定した．

表 2 に 200 byte パケットの最大遅延時間と平均遅延時間

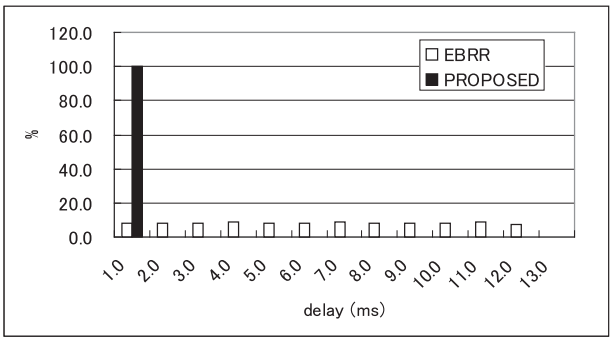

図 5 ケース B) での遅延時間分布（1 ms 間隔）
Fig. 5 Delay distribution of case B) (1 ms interval).

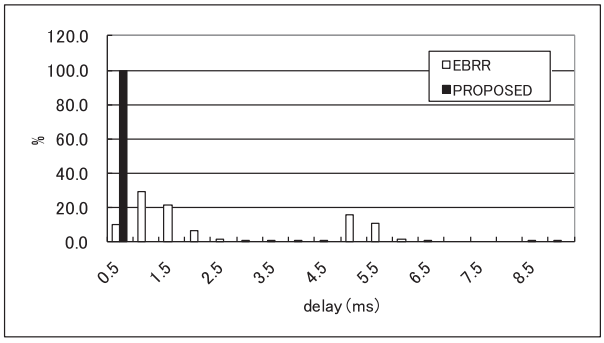

図 6 ケース C) での遅延時間分布（0.5 ms 間隔）
Fig. 6 Delay distribution of case C) (0.5 ms interval).

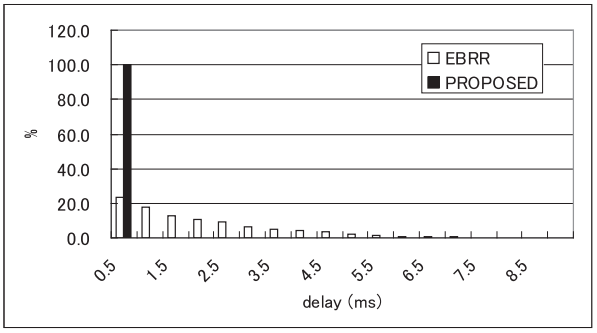

図 7 ケース D) での遅延時間分布（0.5 ms 間隔）
Fig. 7 Delay distribution of case D) (0.5 ms interval).

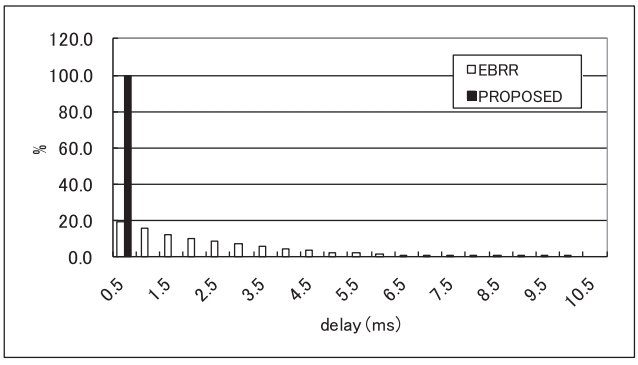

図 8 ケース E) での遅延時間分布（0.5 ms 間隔）
Fig. 8 Delay distribution of case E) (0.5 ms interval).

のシミュレーション結果を示す．表 2 の結果より，EBRR と比較して提案方式では遅延時間が削減されていることが分かる．

図 4，**図 5**，**図 6**，**図 7**，**図 8** に各シミュレーションケースでの遅延時間分布のヒストグラムを示す．この分布より，提案アルゴリズムでの 200 byte パケットの遅延時間は 1 ms 以下であることが分かる．

5. 提案方式の遅延時間上限と公平性の解析

本章では，文献 [3] に示される方法に従い，3 章で述べた提案方式の遅延時間上限と公平性を解析する．解析対象とする遅延時間上限の定義は，文献 [3], [13], [14] の Latency Rate サーバの遅延時間上限の定義とする．この定義は 4 章でシミュレーションを行ったパケットが間欠的にキューに存在する条件とは異なり，つねにすべての flow に送信対象パケットが存在する状態で，quantum 値と全送信レートから決まる各 flow の最低保証帯域値で送信できた場合と比較して，パケット送信時刻が最大どれだけ遅れるかを評価するものとなる．

$\mathrm{flow_i}$ に設定される最低保証帯域値で送信し続けながら，スケジューラ内部に $\mathrm{flow_i}$ のパケットが存在する状態を，$\mathrm{flow_i}$ の busy 期間と呼ぶ．時刻 t_0 から t_1 の間に $\mathrm{flow_i}$ から送信されたデータ量を $D_i(t_0, t_1)$ で表し，$\mathrm{flow_i}$ に設定される最低保証帯域値を r_i で表す．遅延時間上限の定義は，busy 期間の開始時刻を t_0 とし，busy 期間中の任意の時刻を t とした場合に，$D_i(t_0, t) \geqq \max(0, r_i * (t - t_0 - \theta_i))$ を満たす θ_i と定義される．

以下では，全 flow 数を N とし，その中で $\mathrm{flow_1}$ から $\mathrm{flow_M}$ について，パケットサイズが 3 章で導入したパラメータ THRESH 未満となる場合の解析を行う．

5.1 遅延時間上限の解析

遅延時間上限解析のため，次の式 (1) の関係が成立することを最初に示す．

$L_{max,i}$ を $\mathrm{flow_i}$ のパケットの最大サイズ，TH を 3 章で導入したパラメータとする．このとき，スケジューリングラウンド終了時に送信待ちキュー（Queue[i]）にパケットが存在する $\mathrm{flow_i}$ の Credit Counter C_i について，以下の不等式が成立する．

$$\begin{aligned} &TH < C_i \leqq L_{max,i} + TH \quad (1 \leqq i \leqq M) \\ &0 < C_i \leqq L_{max,i} \quad (M < i \leqq N) \end{aligned} \tag{1}$$

・パケットサイズが THRESH 未満の場合

3 章で述べたアルゴリズムは，パケット送信後の C_i の値が TH より大きくなる場合にのみパケットを送信するので，C_i の値は TH より大きくなる．スケジューリングラウンド終了時にパケットがキューに存在し，かつ，C_i の値が $(L_{max,i} + TH)$ より大きい場合，キューの先頭パケット

送信後の C_i の値は TH より大きいので先頭パケットを送信可能となる．先頭パケットが送信されないので，C_i は $(L_{max,i} + TH)$ 以下となる．

・パケットサイズが THRESH 以上の場合

3 章で述べたアルゴリズムは，パケット送信後の C_i の値が 0 より大きくなる場合にのみパケットを送信するので，C_i の値は 0 より大きくなる．スケジューリングラウンド終了時にパケットがキューに存在し，かつ，C_i の値が $L_{max,i}$ より大きい場合，キューの先頭パケット送信後の C_i の値は 0 より大きいので先頭パケットを送信可能となる．先頭パケットが送信されないので，C_i は $L_{max,i}$ 以下となる．

以上より，式 (1) が成り立つ．

次に，遅延時間上限について以下が成り立つことを示す．

---

3 章で示したアルゴリズムの遅延時間上限は，$flow_i$ に設定する最低保証帯域値を r_i，出力インタフェースの送信レートを r と表すと，$1 \leq i \leq M$ の場合の遅延時間上限は $\left(\frac{(Q_i+L_{max,i}+TH)}{r_i} - \frac{M*TH}{r}\right)$ 以下，$M < i \leq N$ の場合の遅延時間上限は $\left(\frac{(Q_i+L_{max,i})}{r_i} - \frac{M*TH}{r}\right)$ 以下となる．

---

3 章のアルゴリズムは，quantum 値 Q_i を各スケジューリングラウンドで Credit Counter C_i に加算する．スケジューリングラウンド x の完了時刻を t_x，t_x 以降の任意の時刻を t とする．そして，スケジューリングラウンド x から t の間に送信された $flow_i$ のパケットデータ量を $D_i(t_x, t)$ で表す．また，スケジューリングラウンド x 完了時の $flow_i$ の Credit Counter C_i の値を $C_i(x)$ で表す．このとき，

$$D_i(th-1, th) = Q_i + C_i(h-1) - C_i(h) \tag{2}$$

が，$D_i(th-1, th) = 0$ の場合を含めて成り立つ．

式 (2) を $h = 1$ から k に変化させた式を加算することで，

$$D_i(t_0, t_k) = k * Q_i + C_i(0) - C_i(k) \tag{3}$$

が得られる．スケジューリングラウンド 0 完了時はパケット未受信で credit counter $C_i = Q_i$ とし，スケジューリングラウンド 0 完了直後に各 flow のパケットを受信したとし，式 (3) に，式 (1) が示す C_i の上限・下限を適用することで，

$$\begin{aligned} &(k+1) * Q_i - L_{max,i} - TH \\ &\leq D_i(t_0, t_k) < (k+1) * Q_i - TH \quad (1 \leqq i \leqq M) \\ &(k+1) * Q_i - L_{max,i} \\ &\leq D_i(t_0, t_k) < (k+1) * Q_i \quad (M < i \leqq N) \end{aligned} \tag{4}$$

が得られる．$t_{k-1} \leq t < t_k$ となる時刻 t について，

$$D_i(t_0, t) = D_i(t_0, t_{k-1}) + D_i(t_{k-1}, t) \geqq D_i(t_0, t_{k-1}) \tag{5}$$

が成り立つ．全 flow の t_0 から t の間の送信パケットデータ量を $D_s(t_0, t)$ と表すと，式 (4) をすべての i について加算することで，

$$\begin{aligned} &(k+1) * \sum_{i=1}^{N} Q_i - \sum_{i=1}^{N} L_{max,i} - M * TH \leq D_s(t_0, t_k) < \\ &(k+1) * \sum_{i=1}^{N} Q_i - M * TH \end{aligned} \tag{6}$$

を得る．各 $flow_i$ に設定する最低保証帯域値を r_i とし，出力インタフェースの送信レートを r とおく．そして，全 flow の最低保証帯域値の合計が出力インタフェースの送信レートを超えないこと，すなわち $r \geq \sum_{i=1}^{N} r_i$ を仮定する．遅延時間上限値解析の前提である全 flow につねに送信待ちパケットが存在することと，3 章に示したアルゴリズム動作より，出力インタフェースではつねにパケットが送信されているので，$D_s(t_0, t) = r * (t - t_0)$ が成り立つ．また，$t_{k-1} < t \leqq t_k$ となる t について $D_s(t_0, t) \leqq D_s(t_0, t_k)$ が成り立つ．これらを式 (6) に代入し，$f = \sum_{i=1}^{N} Q_i$ とおくことで，$t_{k-1} < t \leqq t_k$ となる t について

$$(k+1) > (r * (t - t_0) + M * TH)/f \tag{7}$$

が得られる．式 (7) から得られる k に関する不等式，式 (4) の k を k−1 に置き換えた不等式と式 (5) より，$t_{k-1} < t \leqq t_k$ となる t について，

・$1 \leq i \leq M$ の場合

$$\begin{aligned} &D_i(t_0, t) \geq D_i(t_0, t_{k-1}) \geq k * Q_i - L_{max,i} - TH \\ &> \left(\frac{1}{f}(r(t-t_0) + M*TH) - 1\right) * Q_i - L_{max,i} - TH \end{aligned} \tag{8}$$

・$M < i \leq N$ の場合

$$\begin{aligned} &D_i(t_0, t) \geq D_i(t_0, t_{k-1}) \geq k * Q_i - L_{max,i} \\ &> \left(\frac{1}{f}(r(t - t_0) + M * TH) - 1\right) * Q_i - L_{max,i} \end{aligned} \tag{9}$$

となる．

Q_{min} を Q_i（$i = 1, .., N$）の最小値として，$w_i = Q_i/Q_{min}$ と定義する．f の定義から $f = \sum_{i=1}^{N} Q_i = \sum_{i=1}^{N} w_i * Q_{min}$ なので，

$$w_i = \frac{Q_i}{f} * \sum_{i=1}^{N} w_i \tag{10}$$

となる．また $flow_i$ の quantum 値 Q_i は $flow_i$ の最低保証帯域値 r_i に比例するので，r_i（$i = 1, .., N$）の最小値を r_{min} とすると，

$$r_i = w_i * r_{min} \tag{11}$$

が成り立つ．式 (11) の w_i を式 (10) で置き換えると，

$$r_i = \left(\frac{Q_i}{f} * \sum_{i=1}^{N} w_i\right) * r_{min} \tag{12}$$

を得る．仮定より，各 flow に設定する最低保証帯域値の合計が出力インタフェースの送信レート以下なので，

$$r \geq \sum_{i=1}^{N} r_i = r_{min} * \sum_{i=1}^{N} w_i \quad (13)$$

が成り立つ．式 (13) から得られる r_{min} の上限値を式 (12) に代入することで，

$$r_i \leq r * Q_i/f \quad (14)$$

r は正の値なので，式 (14) から $r_i/r \leq Q_i/f$ となる．これを式 (8)，(9) に代入して整理すると，$t_{k-1} < t \leqq t_k$ となる t について，

・$1 \leq i \leq M$ の場合

$$\begin{aligned} D_i(t_0,t) &> \left(\frac{1}{f}(r(t-t_0)+M*TH)-1\right)*Q_i-L_{max,i}-TH \\ &\geq \left(\frac{r_i}{rQ_i}(r(t-t_0)+M*TH)-1\right)*Q_i-L_{max,i}-TH \\ &= r_i*\left((t-t_0)-\left(\frac{(Q_i+L_{max,i}+TH)}{r_i}-\frac{M*TH}{r}\right)\right) \end{aligned} \quad (15)$$

・$M < i \leq N$ の場合

$$\begin{aligned} D_i(t_0,t) &> \left(\frac{1}{f}(r(t-t_0)+M*TH)-1\right)*Q_i-L_{max,i} \\ &\geq \left(\frac{r_i}{rQ_i}(r(t-t_0)+M*TH)-1\right)*Q_i-L_{max,i} \\ &= r_i*\left((t-t_0)-\left(\frac{(Q_i+L_{max,i})}{r_i}-\frac{M*TH}{r}\right)\right) \end{aligned} \quad (16)$$

となる．よって，$1 \leq i \leq M$ の場合の遅延時間上限は $\left(\frac{(Q_i+L_{max,i}+TH)}{r_i}-\frac{M*TH}{r}\right)$ 以下，$M < i \leq N$ の場合の遅延時間上限は $\left(\frac{(Q_i+L_{max,i})}{r_i}-\frac{M*TH}{r}\right)$ 以下となる．TH が 0 以下であることを考慮すると，パケットサイズが 3 章で導入したパラメータ THRESH 未満の $flow_i$ の遅延時間上限は，そうではない $flow_j$ と比べて，最大パケットサイズが TH の絶対値だけ見かけ上小さくなる効果の分，相対的に小さくなる．

5.2 公平性の解析

Golestani の公平性定義 [15] を拡張した文献 [13] の定義に基づき提案アルゴリズムの公平性 RFB_{ij} を解析する．時刻 t_0 から t の間 2 つの flow（$flow_i$，$flow_j$）が両方 busy 期間である場合に，$flow_i$ の送信データ量を $flow_i$ の最低保証帯域値で割った値と，$flow_j$ の送信データ量を $flow_j$ の最低保証帯域値で割った値の差の上限値をスケジューリングアルゴリズムの公平性と定義する．つまり，

$$|D_i(t_0,t)/r_i - D_j(t_0,t)/r_j| \leqq F$$

を満たす F が公平性を示す．

$D_i(t_0,t)$ の定義から，$t_{k-1} < t \leqq t_k$ となる t について，

$$D_i(t_0,t_{k-1}) \leq D_i(t_0,t) \leq D_i(t_0,t_k) \quad (17)$$

が成り立つ．式 (4) と (17) から，

$$\begin{aligned} &k*Q_i - L_{max,i} - TH \leq D_i(t_0,t) < (k+1)*Q_i - TH \\ &(1 \leqq i \leqq M) \\ &k*Q_i - L_{max,i} \leq D_i(t_0,t) < (k+1)*Q_i \quad (M < i \leqq N) \end{aligned} \quad (18)$$

となる．式 (18) の不等式を用いて，パケットサイズが THRESH 未満の flow と，それ以外の flow の組合せについて公平性を評価する．ここで，任意の i について Q_i が r_i に比例するので，Q_i/r_i は i によらず一定値であることを用いる．

・パケットサイズが THRESH 未満の flow 間の公平性

$$\begin{aligned} &|D_i(t_0,t)/r_i - D_j(t_0,t)/r_j| < \\ &\max(|((k+1)*Q_i - TH)/r_i \\ &\qquad - (k*Q_j - L_{max,j} - TH)/r_j|, \\ &\qquad |((k+1)*Q_j - TH)/r_j \\ &\qquad - (k*Q_i - L_{max,i} - TH)/r_i|) \\ &= \max(|k*(Q_i/r_i - Q_j/r_j) + (Q_i - TH)/r_i \\ &\qquad + (L_{max,j} + TH)/r_j|, \\ &\qquad |k*(Q_j/r_j - Q_i/r_i) + (Q_j - TH)/r_j \\ &\qquad + (L_{max,i} + TH)/r_i|) \\ &= \max(|(Q_i - TH)/r_i + (L_{max,j} + TH)/r_j|, \\ &\qquad |(Q_j - TH)/r_j + (L_{max,i} + TH)/r_i|) \end{aligned}$$

F は $\max(|(Q_i-TH)/r_i+(L_{max,j}+TH)/r_j|, |(Q_j-TH)/r_j+(L_{max,i}+TH)/r_i|)$ となる．

・パケットサイズが THRESH 未満の $flow_i$ とそれ以外の $flow_j$ の間の公平性

$$\begin{aligned} &|D_i(t_0,t)/r_i - D_j(t_0,t)/r_j| < \\ &\max(|((k+1)*Q_i - TH)/r_i - (k*Q_j - L_{max,j})/r_j|, \\ &\qquad |(k+1)*Q_j/r_j - (k*Q_i - L_{max,i} - TH)/r_i|) \\ &= \max(|k*(Q_i/r_i - Q_j/r_j) + (Q_i - TH)/r_i + L_{max,j}/r_j|, \\ &\qquad |k*(Q_j/r_j - Q_i/r_i) + Q_j/r_j + (L_{max,i} + TH)/r_i|) \\ &= \max(|(Q_i - TH)/r_i + L_{max,j}/r_j|, \\ &\qquad |Q_j/r_j + (L_{max,i} + TH)/r_i|) \end{aligned}$$

F は $\max(|(Q_i - TH)/r_i + L_{max,j}/r_j|, |Q_j/r_j + (L_{max,i} + TH)/r_i|)$ となる．

・パケットサイズが THRESH 以上の flow 間の公平性

$$\begin{aligned} &|D_i(t_0,t)/r_i - D_j(t_0,t)/r_j| < \\ &\max(|(k+1)*Q_i/r_i - (k*Q_j - L_{max,j})/r_j|, \\ &\qquad |(k+1)*Q_j/r_j - (k*Q_i - L_{max,i})/r_i|) \\ &= \max(|k*(Q_i/r_i - Q_j/r_j) + Q_i/r_i + L_{max,j}/r_j|, \end{aligned}$$

$$|k*(Q_j/r_j - Q_i/r_i) + Q_j/r_j + L_{max,i}/r_i|)$$
$$= \max(|Q_i/r_i + L_{max,j}/r_j|, |Q_j/r_j + L_{max,i}/r_i|)$$

F は $\max(|Q_i/r_i + L_{max,j}/r_j|, |Q_j/r_j + L_{max,i}/r_i|)$ となる．

いずれの場合でも，F は時間に依存しない値をとる．

6. 既存研究と提案方式の比較

提案方式が拡張対象とした EBRR は SRR に基づくアルゴリズムであるが，EBRR が SRR に対して行ったのと同様の拡張を DRR に対して行ったアルゴリズムとして Aliquem [16] が存在する．提案方式は，(1) 同一スケジューリングラウンドに送信スケジュールされたパケットの中で小さいパケットが先に送信にされるようにする拡張，(2) 大きいパケットが送信スケジュールされるスケジューリングラウンドを後にする拡張，(3) バーストを許容するためクレジットカウンタ値の最大値のパラメータ化の 3 つの拡張から構成される．(2) の拡張については，DRR に基づく Aliquem を採用することで部分的に実現可能である．たとえば図 1 に示すパケット受信シーケンスに対する Aliquem でのパケット送信シーケンス（quantum 値は図 1 と同じ 750 Byte）では，p1-1 はスケジューリングラウンド 2 にスケジュールされ，送信順序は p2-1，p2-2，p1-1，p2-3，p2-4，p2-5 となる．ただし，Aliquem では送信するスケジューリングラウンドが後になるパケットサイズは quantum 値で決まるが，quantum 値は最低保証帯域値と送信のバースト性制御のためのパラメータであり，quantum 値で送信するスケジューリングラウンドが後になるパケットサイズを制御することは適切ではない．提案方式では，quantum 値とは独立なパラメータ TH で，送信するスケジューリングラウンドが後になるパケットサイズを制御できる点が異なる．たとえば 4 章で述べたシミュレーションケース B) の場合，flow ごとの最低保証帯域値の関係から 1,500 byte パケットの flow の quantum 値を 50，200 byte パケットの flow の quantum 値を 4 に設定しており，1 パケット分の Credit Counter 値蓄積に必要なスケジューリングラウンド数は，1,500 byte パケットで 1,500/50 = 30，200 byte パケットで 200/4 = 50 となり，Aliquem では 1,500 byte パケットの方が先のスケジューリングラウンドにスケジュールされる．この場合でも，提案方式では TH = −200 に設定することで，200 byte パケットのスケジューリングラウンドを早めることが可能である．一般的に，遅延時間を減らしたいパケットのサイズが X byte 以下の場合は，THRESH = (X + 1)，TH = −X と設定すればよい．(2) の拡張のみを考慮した場合，提案方式は，パラメータ TH を導入することで EBRR（$TH = -L_{max}$ の場合）と Aliquem（TH = −1 の場合）を一般化したアルゴリズムと考えることができる（厳密には，送信中パケットと，キューで送信待ちのパケットの両方が存在する場合の処理が異なる点はある）．

特定の flow の転送遅延時間を減らすためのパケットスケジューリングアルゴリズムとして Priority Queueing（PQ）が実ネットワークでも用いられている．しかし，提案方式とは異なり，PQ では flow ごとの最低保証送信レート設定等の帯域制御が行えない．また，Web アクセスで要求データ送信後に応答データ受信を行う場合等，1 つの flow の中で，TCP ACK の小さいパケットを送信するフェーズと，データを含む大きいパケットを送信するフェーズが交互に繰り返すケースで，小さいパケットの場合は他の flow の大きいパケットより先に送信するという制御も PQ では実現できないが，提案方式では実現可能である．

7. おわりに

本論文では，計算量が O(1) の性質を保ちながら，1 スケジューリングラウンド内に受信する複数パケットの中で閾値よりも小さいサイズのパケットを先に送信することで，小サイズパケットの遅延時間を減らすことが可能な EBRR の拡張アルゴリズムを提案し，シミュレーションにより有効性の検証を行った．また，遅延時間上限と公平性の解析を行った．今後は，提案方式の通信機器への適用検討を進めてゆく．

参考文献

[1] Shreedhar, M. and Verghese, G.: Efficient Fair Queuing Using Deficit Round-Robin, *IEEE/ACM Trans. Networking*, Vol.4, No.3, pp.375–385 (1996).

[2] Kanhere, S. and Sethu, H.: On the latency Bound of Deficit Round Robin, *Proc. Internation Conference on Computer Communications and Networks*, pp.548–553 (Oct. 2002).

[3] Nikolova, D. and Blondia, C.: Evaluation of Surplus Round Robin Scheduling Algorithm, *Proc. 2006 International Symposium on Performance Evaluation of Computer and Telecommunication Systems* (2006).

[4] Adisshu, H., Parulkar, G. and Varhese, G.: A Relaible and Scalable Striping Protocol, *Proc. ACM SIGCOMM*, pp.131–141 (Aug. 1996).

[5] Kanhere, S. and Sethu, H.: Fair, Efficient and Low Latency Packet Scheduling Using Nested Deficit Round Robin, *Proc. IEEE Workshop on High Performance Switching and Routing*, pp.6–10 (2001).

[6] Kanhere, S., Sethu, H. and Parekh, A.: Fair and Efficient Packet Scheduling Using Elastic Round Robin, *IEEE Trans. Parallel and Distributed Systems*, Vol.13, No.3, pp.324–336 (2002).

[7] Nikolova, D. and Blondia, C.: Last-Backlogged First-Served Deficit Round Robin (LBFS-DRR) Packet Scheduling Algorithm, *International Coference on Networks*, pp.330–335 (Nov. 2007).

[8] Lenzini, L., Mingozzi, E. and Stea, G.: Eligibility-Based Rount Robin for Fair and Efficient Packet Scheduling in Wormhole Switching Networks, *IEEE Trans. Parallel and Distributed Systems*, Vol.15, No.3, pp.244–256 (2004).

[9] Nichols, K., Blake, S., Baker, F. and Black, D.:

RFC2474: Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers, available from ⟨http://www.rfc-editor.org/rfc/rfc2474.txt⟩ (Dec. 1998).
[10] Matsuda, T., Horiuchi, E. and Yokotani, T.: A Proposal of a New Packet Scheduling Algorithm Which Can Reduce the Delay of Small Packets, *Proc. IEEE GCCE*, pp.266–267 (2012).
[11] Matsuda, T.: A Proposal of a New Packet Scheduling Algorithm and Its Evaluation, *Proc. ITU Kaleidoscope* (2013).
[12] available from ⟨https://www.ntt-east.co.jp/databook/pdf/2010_09-10.pdf⟩.
[13] Stiliadis, D.: Traffic Scheduling in Packet-Switched Networks: Analysis, Design, and Implementation, Ph.D. Dissertation, University of California at Santa Cruz, USA (1996).
[14] Stiliadis, D. and Varma, A.: Latency-Rate Servers: A General Model for Analysis of Traffic Scheduling Algorithms, *IEEE/ACM Trans. Networking*, Vol.6, No.5, pp.611–624 (1998).
[15] Golestani, S.: A self-clocked fair queueing scheme for broadband applications, *Proc. INFOCOM '94*, pp.636–646 (1994).
[16] Lenzini, L., Mingozzi, E. and Stea, G.: Aliquem: A Novel DRR Implementation to Achieve Better Latency and Fairness at O(1) Complexity, *Proc. 10th Int. Workshop on Quality of Service* (*IWQoS*), pp.77–86 (May 2002).

松田 哲史 （正会員）

昭和 60 年東京大学工学部電気工学科卒業，昭和 62 年同大学大学院工学系研究科修士課程修了．現在，三菱電機（株）に勤務．IP 通信機器および IP ネットワークの研究開発に従事．電子情報通信学会，IEEE 各会員．