高速バックボーンネットワークにおける 公平性を考慮した階層化パケットスケジューリング方式

牧 一之進† 下西 英之‡ 村田 正幸† 宮原 秀夫†

† 大阪大学 大学院基礎工学研究科
〒 560-8531 大阪府豊中市待兼山町 1-3
Phone: 06-6850-6588, Fax: 06-6850-6589
{i-maki, murata, miyahara}@ics.es.osaka-u.ac.jp

‡NEC ネットワーキング研究所
〒 216-8555 神奈川県川崎市宮前区宮崎 4 丁目 1-1
Phone: 044-856-2123, Fax: 044-856-2230
simonisi@ccm.cl.nec.co.jp

あらまし フロー間の公平性を実現するためには、すべてのルータにおいてフロー毎の制御を行うことが望ましいが、高速なパケット処理を必要とするコアルータにおいては容易なことではない。そこで、本稿では大規模ネットワークにおいて、フロー毎の優れた公平性を実現するため、低速なエッジルータから高速なコアルータまで実装可能なスケーラブルなパケットスケジューリング方式を提案する。本方式では低速なルータではフロー毎にキューを割り当てるが、高速なルータでは複数のフローを集約してキューを割り当てる。フローの集約を行う場合でも、各キューに収容されたフローの本数を推定して、その本数に比例した帯域を割り当て、さらに同じキュー内でレートの高いフローを発見して廃棄制御も行うことにより可能な限り公平性を向上させる。本稿では、シミュレーションによって提案方式の評価を行い、フローを集約した制御を行っても高い公平性を実現できることを示す。
和文キーワード 公平性、 スケーラビリティ、 スケジューラ、 QoS、 シミュレーション

Hierarchically Aggregated Fair Queuing (HAFQ) for Per-flow Fair Bandwidth Sharing in High-speed Backbone Networks

Ichinoshin Maki† Hideyuki Shimonishi‡ Masayuki Murata† Hideo Miyahara†

†Graduate School of Engineering Science,
Osaka University
1-3 Machikaneyama, Toyonaka,
Osaka 560-8531, Japan
Phone: +81-6-6850-6588, Fax: +81-6-6850-6589
{i-maki, murata, miyahara}@ics.es.osaka-u.ac.jp

‡Networking research laboratories,
NEC corporation
1-1 Miyazaki 4-Chome, Miyamae-ku, Kawasaki,
Kanagawa, 216-8555
Phone: +81-44-856-2123, Fax: +81-44-856-2230
simonisi@ccm.cl.nec.co.jp

Abstract It is promising to allow per-flow queue management in all routers in order to realize per-flow fair service in backbone networks. However, it is not easy to allow per-flow queue management in core routers, requiring to support many flows. In this paper, we propose a scalable queue management scheme according to the forwarding speed of line interfaces for realizing per-flow fair service. It allows a scalable queue management; per-flow queue management in edge routers and flow aggregation in core routers. Against the aggregated flows, the proposed scheme estimates the number of flows and allocates bandwidth in proportional to the estimated number of flows. Further it finds the flow obtaining for achieving higher throughput and preferentially drops the packets of that flow. We evaluate the proposed scheme through extensive simulation studies.
key words fairness, scalability, scheduler, QoS, simulation

1 初めに

現在、インターネットトラヒックの大部分を占める Best effort 系トラヒックに関する重要な課題の一つとして、各ユーザへの公平なサービスの実現がある。今後、各ユーザのアクセス帯域が大きくなれば特定のユーザのみが大きな帯域を占有することも有り得るため、ユーザ間の公平なサービスはますます重要になってくると考えられる。

フロー毎の公平性を実現するパケットスケジューリング方式としては、従来から数多くの研究がなされている。一般的にエッジルータは比較的低速であり、扱うフローの数も少ないのでフロー毎の情報を持つことができる。CSFQ [1, 2] では、エッジではフロー毎のレートを測定してパケットヘッダに書き込んでおき、コアではパケットヘッダに書かれたレートによって、動的にそれぞれのフローの廃棄確率を決定する方式を提案している。ところが CSFQ ではヘッダの拡張が必要であり、ネットワークのすべてのエッジルータを更新する必要があることが問題である。

ヘッダの拡張を行わない方式として DRR [3] がある。この方式ではフロー毎にキューを設けてスケジューリングを行うことにより、フロー毎の公平性を実現する。ところがこの方式ではすべてのフローに対してキューを設けるため、数多くのフローを高速に扱うバックボーンのコアルータへは実装困難である。また FRED [4] では、それぞれのフローのバッファの使用率に応じて廃棄確率を決定するため、やはりコアルータへの適用は困難である。

このように CSFQ のようなヘッダの拡張の必要がなく、かつ DRR や FRED のようなフロー毎の情報を持たない方式が望ましい。そこで我々の提案する方式では、基本的に DRR のような per-flow のスケジューリングを行うが、扱うべきフロー数にしたがってフローの集約を行うことで、エッジルータからコアルータまで適用できるスケーラブルなスケジューリング方式を提案する。フローが複数集約されたキューでは収容するフローの数を推定し、その推定された数に比例した帯域を割り当てる。さらにフローの数を推定する際に、同じキュー内でより多くの帯域を使用しているフローを検出することが可能であるので、これを利用した廃棄制御も行う。

また per-flow 制御を行う場合には、アクティブなフローを 1 本ずつ識別してキューを割り当てる必要があり実装が難しい。これに対して提案方式では、複数のフローを同一のキューに集約する場合にも、適当なハッシュ関数を用いてキューを割り当てるだけで良いため、実装も容易である。

以下、2 章では提案方式について説明し、3 章ではシミュレーションによって、従来方式と提案方式を比較して提案方式の有効性を示す。最後に 4 章でまとめを述べる。

2 HAFQ (Hierarchically Aggregated Fair Queuing)

2.1 提案方式の概略

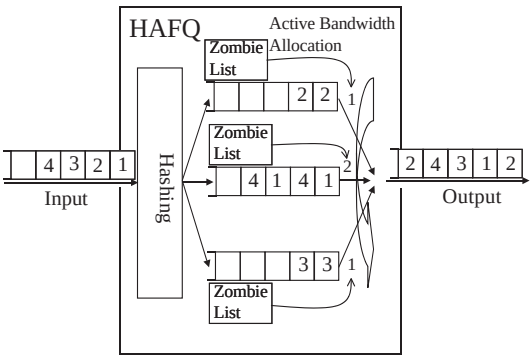


図 1: 提案方式の概略図

提案方式の概略図を図 1 に示す。あるフローのパケットがルータに到着すると、まずヘッダ情報からハッシュ関数をひいて、パケットをどのキューに収容するか決定する。このとき各キューへの振り分けは、それぞれに収容されるフローの数ができる限り均等になるように行うのがよい。文献 [5] において、CRC16 がフローをバランス良く分配できることが示されているため、我々はハッシュ関数として CRC16 を採用した。

ルータに到着したパケットにキューが割り当てられると、各キューのゾンビリストを用いて、そのキューに収容されているフローの数を推定し、推定されたフローの数に比例した帯域を各キューに割り当てる。ゾンビリストとは過去に到着したフローに関する履歴を保持しているリストであるが、全フローの情報を保持しておく必要はない。さらにゾンビリストはレートの高いフローの検出を行うことも可能であるので、同じキュー内でレートの高いフローのパケットを優先的に廃棄することにも用いる。

2.2 ゾンビリスト

ゾンビリストは一定サイズのテーブルであり、それぞれのエントリとして、{Flow ID、カウンタ } を 1 つの組として持っている。これをゾンビと呼ぶ。パケットがルータに到着すると以下のような動作を行う。

- ゾンビリストを全て検索する
  - 到着して来たパケットの Flow ID がリスト内に存在すれば、カウンタ値を 1 つ増やす。この動作を Hit と呼ぶことにする。
  - 存在しなければ、リスト内からランダムに 1 つ選択し、
    - 確率 q で Flow ID を到着して来たパケットの Flow ID で置き換え (Swap と呼ぶことにする)、カウンタ値を 1 に初期化する。
    - 確率 $1 - q$ で何も行わない。このとき No-Swap と呼ぶ。

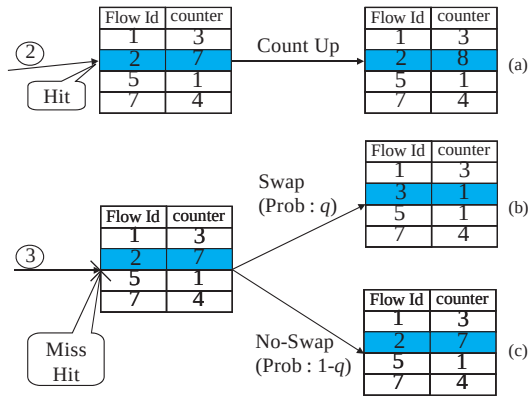


図 2: ゾンビリスト

例えば図 2 のように、Flow ID が 2 であるパケットが到着すると Hit となり、その Flow ID が保持されているエントリのカウンタ値を 1 増やす (図 2(a))。また、Flow ID が 3 であるパケットが到着した場合、リスト内を検索してもエントリが見つからないので、リスト内からランダムに 1 つのエントリを選択し、q の確率で到着して来たパケットの Flow ID で置き換え、そのエントリのカウンタ値を 1 に初期化する (図 2(b))。このとき 1-q の確率で何も行わない (図 2(c))。

SRED/ZLRED [6, 7] でもゾンビリストを用いて、レートの高いフローを発見する方式が提案されている。これらの方式では、ゾンビリストを全て検索するのではなく、1 つもしくは少数のゾンビのみを検索するだけである。そのためゾンビリスト内に同じ Flow ID をもつゾンビが存在する可能性があり、ゾンビリストを効率良く使用することができない。ゾンビリストが大きいときにはリストを全て検索するオーバヘッドがかかるが、提案方式はゾンビリストを大きくするよりも、ゾンビリストを小さくしてキュー数を多くした方がよいと考え、ゾンビリストのエントリ数を極力少なくするので、このようなことが可能である。

2.3 フロー数の推定方法

SRED では、ゾンビリストを用いてフローの本数を推定する方式が述べられている。この方式では全てのフローがほぼ同じレートでルータに到着しているときに限り、その推定フロー数は実際のフロー数に近い値になるが、レートの高いフローが存在すれば、その推定値は正しいフロー数よりも小さな値となってしまう。

そこで我々は、レートが不均一であるような場合でも、正確にフロー数を推定できる方式を提案する。本方式では、各到着フローのレートを推定し、それからこれを平均して全フローの平均レートを推定し、これをリンク容量で割ったものがフローの本数であると推定する。すなわち、全フロー数を N、フロー i のレートを λ_i、全フローの平均レートを λ_{avg} とすると、$\lambda_{avg} = \sum_{i=1}^{N} \lambda_i / N$ となるため、

$$N = \frac{\sum_{i=1}^{N} \lambda_i}{\lambda_{avg}} \tag{1}$$

となる。全到着レートを平均レートで割ったものを推定フロー数とするため、本方式ではレートにかたよりがある場合でも正しくフロー数を推定することができる。以下では、全到着レートに占めるフロー i のレートの割合を R_i として、この R_i の平均 R_{avg} から推定フロー数を求める。また本稿ではパケット長を固定と仮定とするが、可変長への拡張も容易である。

フローの本数を推定するにあたって、まず R_i の推定値 $\widetilde{R_i}$ を求める。$\widetilde{R_i}$ はゾンビの内容が別のフローの ID で置き換えるときに計算を行う。これはこのときのカウンタ値が、そのフローのレートに比例した値となっているためである。

今、あるゾンビ $j\,(1 \le j \le M)$ に着目する。M はゾンビリストの大きさであり、このゾンビ内に保持されているフローの ID を $X_j = i$、R_{X_j} をフロー X_j の到着割合であるとする。フロー i のパケットが到着し、あるゾンビがこのパケットの ID で置き換えられてカウンタ値を 1 とし、カウンタ値が 1 のまま、他のフローに置き換えられてしまう確率、すなわちカウンタの最大値が 1 である確率を P_1 とすると、

$$\begin{aligned} P_1 &= (1-R_i)a + (1-R_i)(1-a)(1-R_i)a \\ &\quad + \{(1-R_i)(1-a)\}^2(1-R_i)a \\ &\quad + \cdots + \{(1-R_i)(1-a)\}^n(1-R_i)a \\ &= \frac{(1-\sum_{j=1}^{M} R_{X_j})\frac{q}{M}}{(1-\sum_{j=1}^{M} R_{X_j})\frac{q}{M} + R_i} \end{aligned}$$

となる。このとき a は到着パケットがフロー i でないときに置き換えが発生する確率であり、q は置き換え確率であるとする。同様にカウンタ値が n までカウントされて、他のフローの ID で置き換えられてしまう確率 P_n は、

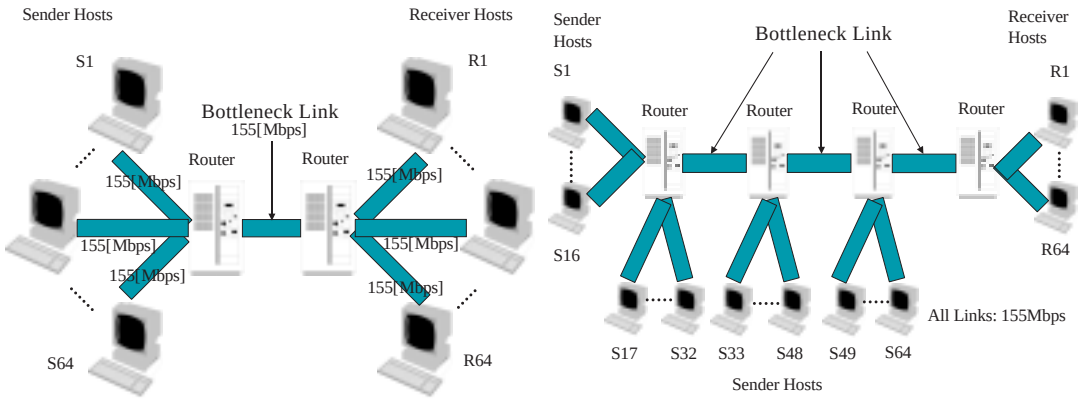
$$\begin{aligned} P_n &= R_i^{n-1}P_1 + (1-R_i)(1-a)R_i^{n-1}P_1 \\ &\quad + \{(1-R_i)(1-a)\}^2 R_i^{n-1}P_1 \\ &\quad + \cdots + \{(1-R_i)(1-a)\}^n R_i^{n-1}P_1 \\ &= \frac{R_i^{n-1}(1-\sum_{j=1}^{M} R_{X_j})\frac{q}{M}}{\{(1-\sum_{j=1}^{M} R_{X_j})\frac{q}{M} + R_i\}^n} \end{aligned}$$

となる。それゆえ、フロー i のカウンタ最大値の期待値 E_i は以下のようになる。

$$\begin{aligned} E_i &= P_1 + 2P_2 + 3P_3 + \cdots + \infty \\ &= \frac{(1-\sum_{j=1}^{M} R_{X_j})\frac{q}{M} + R_i}{(1-\sum_{j=1}^{M} R_{X_j})\frac{q}{M}} \end{aligned} \tag{2}$$

したがって式 (2) より、ゾンビが置き換えられたときカウンタ値が E_i であれば、そのフロー i の到着割合の推定値 $\widetilde{R_i}$ は、

$$\widetilde{R_i} = \left(1 - \sum_{j=1}^{M} R_{X_j}\right) \frac{q}{M}(E_i - 1)$$



(a) シングルリンクモデル　　(b) 多段接続モデル

図 3: ネットワークモデル

となる。ここで $\sum_{j=1}^{M} R_{X_j}$ を平均ヒット率 p と近似すると、

$$\widetilde{R_i} = (1-p)\frac{q}{M}(E_i - 1)$$

となる。このようにゾンビが置き換えられるごとに $\widetilde{R_i}$ を求め、その平均をとると平均到着割合が求まる。しかしレートの高いフローのパケットは、他のフローのパケットに比べて数多く到着し、そのフローのレートは他のフローのレートよりも多く平均到着割合に組み入れられるため、レートが不均一であれば平均レートは大きめに推定される。ここで到着割合 R_i のパケットは、単位時間当たり R_i/E_i 回ゾンビに新しく登録されるので、到着割合の平均をとる際には、$(\widetilde{R_i}/E_i)^{-1}$ で重みをつけて平均をとれば良いことになる。すなわち、

$$R_{avg} = \left\{1 - \beta\left(\frac{E_i}{\widetilde{R_i}}\right)\right\} R_{avg} + \beta\left(\frac{E_i}{\widetilde{R_i}}\right)\widetilde{R_i}$$

である。このときあるキューに収容されている推定フロー数は、$\sum_{i=1}^{N} \widetilde{R_i} = 1$ と式 (1) を利用すると $\dfrac{1}{R_{avg}}$ となる。本方式ではフロー数 $\leq$ エントリ数の場合には、定常状態でカウンタ値が発散してしまうため、フロー数 $>$ エントリ数の場合にのみ有効である。そこで我々は常にフロー数 $>$ エントリ数となるように、エントリ数を動的に減らしていくことでこの問題を解決するが、紙面の制限上ここでは述べない。

2.4 カウンタによるパケット廃棄

提案方式では各キュー間の公平性は実現できるが、同一キュー内でのフロー間の公平性は実現できない。そこで、ゾンビリストを用いて、同じキュー内でより多くの帯域を使用しているフローを発見し、そのフローのパケットを優先的に廃棄することで、キュー内での公平性を向上させる。

2.2 節で述べたように、あるゾンビのカウンタ値は、ある期間内においてルータに到着してきたパケットの数となる。このカウンタ値が大きければ大きいほど、そのフローは他のフローよりも多くの帯域を使用している可能性が高い。そこで、あるフローのカウンタ値が、全ゾンビのカウンタ値の平均よりも大きい場合には、このフローのパケットの廃棄優先度を上げ、キュー長が半分以上のときに、廃棄優先度の高いパケットを廃棄する。

3 性能評価

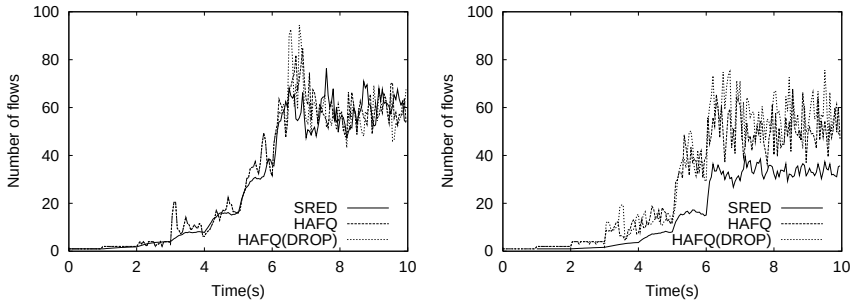
3.1 評価モデル

本稿で用いるネットワークモデルはシングルリンクモデル (図 3(a)) および多段接続モデル (図 3(b)) とする。アクセスリンクの帯域ならびにボトルネックリンクの帯域は等しく 155 Mbps とし、伝搬遅延時間はアクセスリンクでは 0.1 ms、ボトルネックリンクでは 1 ms とする。また、それぞれの端末は TCP (Tahoe) あるいは UDP (3.2 Mbps の固定レート) を用いて送信を行うものとする。このとき端末の送信するデータは無限にあるとする。シミュレーションには NS シミュレータ [8] を用いた。

3.2 推定フロー数の検証

提案するフロー数推定方式の有効性を示すために、SRED のフロー数推定方式との比較を行う。また、カウンタによる廃棄がフロー数推定に与える影響についても評価する。シングルリンクモデルを用い、その端末からは送信時刻が 0 [s] から 1 秒毎に、そのフローの本数が 2 倍になるように、つまり、1 本、2 本、4 本、$\cdots$ となるように送信を開始する。全てのフローが同一のキューに収容されることとし、このキューにおける推定フロー数を評価する。ゾンビリストのエントリ数は 4 とする。評価の対象として、同一環境下にある TCP のみが存在する場合、TCP と UDP が混在する場合 (時刻 0 [s] では TCP1 本のみとし、以降は TCP,UDP が半分ずつ存在する) の評価を行う。HAFQ はカウンタによる廃棄を行わない方式を表し、HAFQ (DROP) はカウンタによる廃棄を行う方式であるとする。

SRED でも全ての TCP フローが同一環境下にあれば、



(a) 同一環境下にある TCP　　(b) TCP と UDP が混在する場合

図 4: SRED と提案方式の推定フロー数

その推定フロー数は実際のフロー数に近いものとなることが分かる。提案方式の場合でも同様である (図 4(a))。ところが、UDP が混在する場合、SRED では推定されたフロー数が実際の本数とはかけはなれた 32 本付近となっていることが分かる。これは UDP のレートが TCP のレートに比べて高いために、あたかも UDP だけが存在して、TCP が存在していないかのように見えるからである。提案方式では推定フロー数が大きく改善されており、カウンタによる廃棄を行った場合にはそれぞれのフローのレートがより均一になるので、推定フロー数は実際のフロー数 64 に近づき、平均すると 60 付近と推定された (図 4(b))。

3.3　スループットの評価

シングルリンクモデルおよび多段接続モデルにおける、FIFO を用いた場合と提案方式を用いた場合の各フローの平均スループットを示し、フロー間のスループットのばらつきを評価する。全フロー数は 64 とし、各フローは時刻 0 [s] に同時に送信を開始するものとする。HAFQ ではフローの各キューへの振り分けは CRC16 によるハッシュ法であるとし、キュー数を 64、各キューのゾンビ数を 4 とする。

TCP が同一環境下にある場合、FIFO ではスループットのばらつきが大きいことが分かる。これに対して提案方式は高い公平では一部のフローを除いて高い公平性が得られていることが分かる (図 5(a))。次に UDP が混在する場合を考える。ここでは Flow ID が 33∼64 であるものは UDP であるとする。提案方式では特に TCP フローと UDP フローの間の公平性が改善し、カウンタによる廃棄を行った場合には、さらに大きく改善していることが分かる。Flow ID が 10 と 50 のように同一のキューに TCP と UDP が収容された場合、HAFQ ではレートの高いフローに対して制御を行っていないため、スループットの差が大きくなってしまう。これに対して HAFQ (DROP) では、このフローをゾンビリストを用いて発見し、優先的にパケットを廃棄しているので、両者のスループットの差が小さくなっている (図 5(b))。

3.4　フロー数を変化させた場合の評価

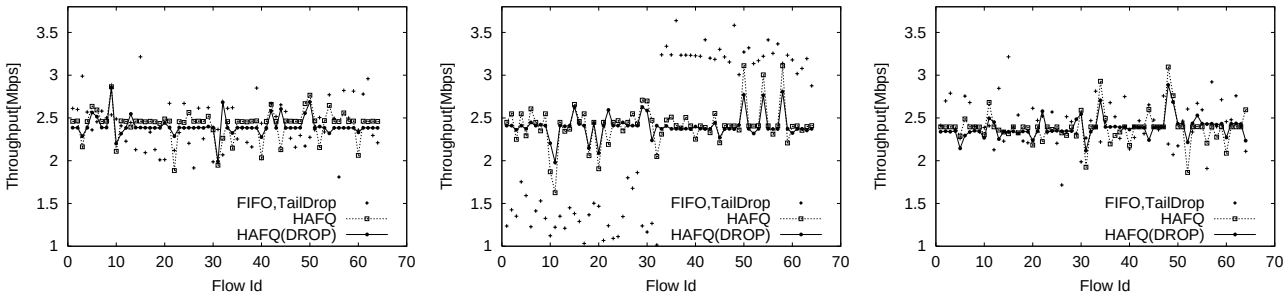
フロー数を変化させたときの公平性の変化について評価し、提案方式が FIFO や DRR に比べてフロー数が多い場合に特に有効であることを示す。ここでは公平性の尺度として Fairness Index を用いる。Fairness Index (f) は以下の式のように、各フローのスループット (フロー i のスループットを x_i とする) のばらつきが小さい場合にはその値は 1 に近づき、逆にばらつきが大きい場合にはその値は 1 から遠ざかる。

$$f(x_1, x_2, x_3, \cdots, x_N) = \frac{(\sum_{i=1}^{N} x_i)^2}{N \sum_{i=1}^{N} {x_i}^2}$$

ここでは DRR、HAFQ ともにキュー数を 64 とする。DRR の場合、フローの本数が 64 本までは完全な per-flow キューイングを行い、フローの本数が 64 本を越えたあとはフロー毎にランダムに収容されるキューを設定する。一方 HAFQ ではハッシュ関数を用いてキューの割り当てを行う。ゾンビ数は 4 とする。FIFO では図 6 のように、フロー数を多くしていくとフロー間の公平性は悪くなっていくことが分かる。DRR ではフロー数が 64 本に至るまでは、per-flow キューイングを行っているため Fairness Index は 1 を示しているが、フロー数が 64 本を越えると各キューに収容されているフローの数にばらつきが出るため、公平性は低下していく。提案方式では、フロー数が少ないときには DRR と同等の性能を示すことが分かった。さらにフロー数が多くなると、各フローの公平性はしだいに低下していくものの、その有効性は従来の手法に比べて高いといえる。またカウンタによる廃棄を行った場合、フロー数が多いときの公平性の低下を抑えることができる。特に UDP が混在する場合、レートの高い UDP フローのパケットを優先的に廃棄することで、レートのばらつきが抑えられ公平性が向上した。

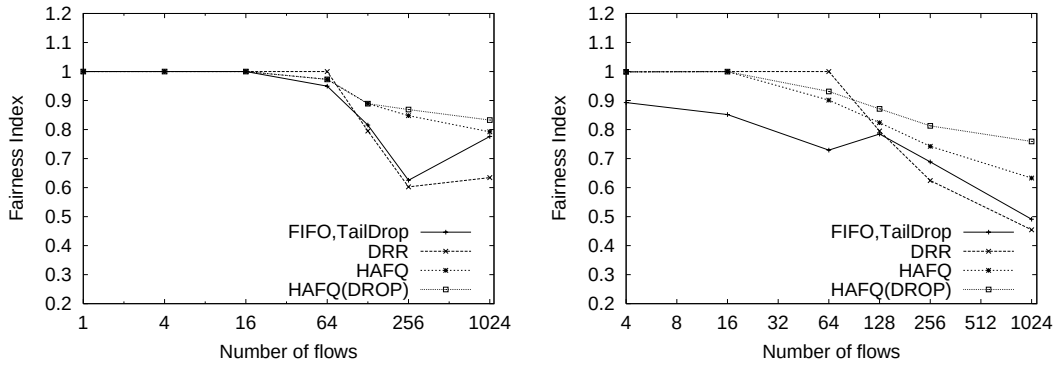
4　まとめ

本稿では、 フロー毎の優れた公平性を実現し、エッジルータやコアルータの能力に合わせたスケーラブルに実装可能なスケジューリング方式の提案を行った。提案方式は各キュー毎に収容されているフローの数を推定し、



(a) 同一環境下にある TCP　(b) TCP と UDP が混在する場合　(c) 多段接続モデル

図 5: 各フローのスループット



(a) TCP のみの場合　(b) TCP と UDP が混在する場合

図 6: フロー数を変化させた場合の Fairness Index

その数に比例した帯域を各キューに割り当てることで、フロー間の公平性を実現する。さらに同じキュー内で、より多くの帯域を使用しているフローを発見して、そのフローのパケットを優先的に廃棄することで、フロー間の公平性をさらに向上させることができた。

参考文献

[1] R. Kapoor, C. Casetti, and M. Gerla, “Core-stateless Fair Bandwidth Allocation for TCP flows,” in *Proceedings of the IEEE ICC2001*, June 2001.

[2] I. Stoica, S. Shenker, and H. Zhang, “Core-Stateless Fair Queueing: Achieving Approximately Fair Bandwidth Allocations in High Speed Networks,” in *Proceedings of the ACM SIGCOMM’98*, pp. 118–130, October 1998.

[3] M. Shreedhar and G. Varghese, “Efficient Fair Queueing using Deficit Round Robin,” *IEEE/ACM Transactions on Networking*, vol. 4, no. 3, pp. 375–385, 1996.

[4] D. Lin and R. Morris, “Dynamics of Random Early Detection,” *ACM Computer Communication Review*, vol. 27, pp. 127–137, October 1997.

[5] Z. Cao, Z. Wang, and E. Zegura, “Performance of Hashing-Based Schemes for Internet Load Balancing,” in *Proceedings of IEEE INFOCOM 2000*, pp. 332–341, 2000.

[6] L. W. Teunis J. Ott, T.V. Lakshman, “SRED: Stabilized RED,” in *Proceedings of IEEE INFOCOM 1999*, March 1999.

[7] 倉田 謙二, 長谷川 剛, 村田 正幸, “TCP バージョン間の公平性の向上のための RED の改善方式に 関する検討,” **電子情報通信学会 技術研究報告** *(SSE00-06)*, pp. 13–18, June 2000.

[8] “UCB/LBNL/VINT Network Simulator - ns (version 2).” available at `http://www-mash.cs.berkeley.edu/ns/`.