社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

TCP コネクションのためのアクセスリンク資源管理方式

東　　和弘†　　長谷川　剛††　　村田　正幸††

† 大阪大学 大学院情報科学研究科　〒 560–8531 大阪府豊中市待兼山町 1–3
†† 大阪大学 サイバーメディアセンター　〒 560-0043 大阪府豊中市待兼山町 1–32
E-mail: †k-azuma@ist.osaka-u.ac.jp, ††{hasegawa,murata}@cmc.osaka-u.ac.jp

あらまし　現在のインターネットでは DSL などの普及によって、ユーザホストとインターネットをつなぐアクセスリンク帯域は増加している。しかし、依然としてその容量は十分ではなく、ユーザが複数のネットワークアプリケーションを同時に利用するような場合には資源競合が発生し、ボトルネックになりやすい。また、標準の TCP コネクションのスループットは各コネクションのラウンドトリップ時間などのパラメータに大きく影響されるため、複数のアプリケーションを実行している状況ではアクセスリンク帯域は必ずしもそれらの特徴を考慮して共有されない。そこで、本稿ではボトルネックとなるアクセスリンク資源を効率的に活用するためのアクセスリンク資源管理方式を提案する。提案方式においては、既存の TCP プロトコルに改変を加えることなく、ユーザホスト上で全ての TCP コネクションに割り当てる受信バッファの総量を調整し、QoS を考慮して受信バッファを各 TCP コネクションに割り当てる。シミュレーションによる性能評価結果より、提案方式はデータ転送時間の減少、およびアクセスリンクでの輻輳の回避や遅延の減少に大きな効果があることを示す。
キーワード　TCP、アクセスリンク、受信バッファ、公平性

Receiver-based Management Scheme of Access Link Resources for QoS-Controllable TCP Connections

Kazuhiro AZUMA†, Go HASEGAWA††, and Masayuki MURATA††

† Graduate School of Information Science and Technology, Osaka University
1–3 Machikaneyama, Toyonaka, Osaka 560–8531, Japan
†† Cybermedia Center, Osaka University
1–32 Machikaneyama, Toyonaka, Osaka 560–0043, Japan
E-mail: †k-azuma@ist.osaka-u.ac.jp, ††{hasegawa,murata}@cmc.osaka-u.ac.jp

Abstract　Although the bandwidth of access networks are rapidly increasing with the latest techniques such as DSL and FTTH, the access link bandwidth remains a bottleneck especially when users activate multiple network applications simultaneously. Furthermore, since the throughput of a standard TCP connection is dependent on various network parameters including round trip time and packet loss ratio, the access link bandwidth not shared among the network applications according to their characteristics. We present management scheme of the access link resource for effective utilization of the access link bandwidth and control of TCP connection's throughput. Our proposed scheme adjusts the total amount of the receive socket buffer assigned to TCP connections to avoid the congestion at the access network, and assigns it to each TCP connection according to their characteristics in consideration of QoS. The simulation results show that our proposed scheme has some great effects in the reduction of the data transfer time, the avoidance of the congestion at the access link and the decrease of the delay, compared with the traditional TCP and some related works.
Key words　TCP, Access link, Receive buffer, Fairness

1. はじめに

インターネットの急速な発展にともなうトラヒックの増大に対し、バックボーンネットワークでは広帯域化、高速化が急速に進められている。その結果、現在のインターネットにおいてはエンドホスト資源やアクセスリンク資源がボトルネックになりつつある。たとえば、繁忙な Web サーバなどにおいて TCP を用いたデータ転送を行う際、エンドホストのソケットバッファ、ディスクリプタ、CPU 資源などの TCP コネクションを確立するための資源が不足することによってエンドホストがボトルネックなることが問題となる。そこで、我々の研究グループでは、この問題を解決するためにエンドホストにおける TCP コネクション資源の管理方式を提案し、シミュレーション、実装実験を通して、その有効性を確認した [1]。一方、現在のインターネットでは DSL (Digital Subscriber Line) などの普及によって、ユーザホストとインターネットを接続するアクセスリンク帯域は増加している。しかしながら、依然としてアクセスネットワークの帯域はバックボーンネットワークに比べると十分ではなく、特に図 1 のようにユーザが複数のネットワークアプリケーションを同時に利用するような場合では資源競合が発生し、アクセスリンク帯域がボトルネックになりやすい。また、標準の TCP コネクションのスループットはラウンドトリップ時間 (RTT) などのパラメータに大きく影響されるため [2]、必ずしもアプリケーションの特徴を考慮した割合でアクセスリンク帯域がそれらのアプリケーション間で共有されない。たとえばバルクデータ転送を行う FTP などで用いられる TCP コネクションのスループットを高くして、それほど高いスループットが要求されず、バックグラウンドで動作するのが望ましいネットワークアップデートなどで用いられる TCP コネクションのスループットを制限する、といった制御を行うことはできない。また、P2P、FTP などのアプリケーションを用いてバルクデータ転送を行っているときに、新たに Web ドキュメントの転送を開始すると、既存の TCP コネクションがアクセスリンク帯域を使い切っている場合、輻輳が発生する。この輻輳によって、新たに開始された Web ドキュメント転送で用いられる TCP コネクションでパケットが廃棄されると、高い確率でタイムアウトが発生し、データ転送時間は大きく増加する [3, 4]。

そこで本稿では、これらの問題点を解決し、ボトルネック

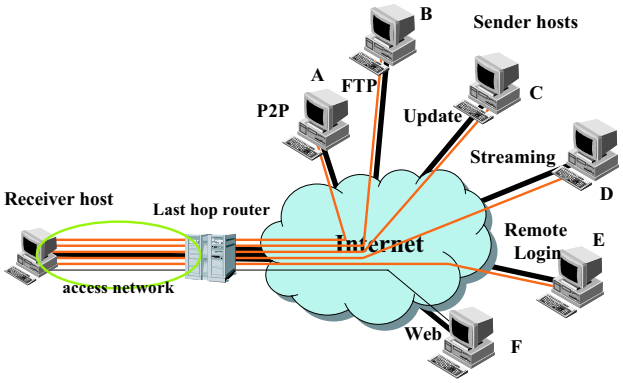


図 1　アクセスネットワークの輻輳

となるアクセスリンク資源を有効に活用するために、ユーザホストで全ての TCP コネクションに割り当てられる受信バッファの総量を仮想的に調整し、各 TCP コネクションへの受信バッファの割り当てを決定する方式を提案する。提案方式において受信バッファの総量を仮想的に調整するのは、アクセスリンクへのパケットの到着レートを抑えることにより、アクセスリンクでの輻輳を回避するためである [5,6]。提案方式では、各コネクションに割り当てる受信バッファサイズを決定する際、それぞれのコネクションの性質を考慮する。すなわち、ユーザホスト上で処理する TCP コネクションを、Web ドキュメント転送のように転送データ量が少なく生存期間の短い TCP コネクション (short-lived コネクション) と、FTP を用いたバルクデータ転送のように転送データ量が多く生存期間の長い TCP コネクション (long-lived コネクション) の 2 種類に分ける。上述した通り short-lived コネクションでは、パケット廃棄によってデータ転送時間が大きく増加するため、そのパケットを優先的に処理し、パケットがアクセスリンクを通ってユーザホストまで到着する割合を向上させる必要がある。これに対し、long-lived コネクションでは、ボトルネックとなるアクセスリンク資源を有効に活用するために、上述したようなアプリケーションの特徴を反映した QoS を考慮する必要がある。つまり提案方式においては、short-lived コネクションを優先的に処理しつつ、QoS を考慮して long-lived コネクション間でアクセスリンク帯域を共有させることを目標とする。

本稿で提案するアクセスリンク資源管理方式は、受信側ホストで動作する。これは、標準の TCP の輻輳制御機構では、送信側ホストにおいて輻輳制御の多くが行われるため、受信側ホストに近いアクセスリンク帯域を正確に推測することはできないためである。また、各送信側ホストは独立した輻輳制御を行っているため、受信側ホストにおいて動作している他のネットワークアプリケーションの特徴を反映させることができないことからも、受信側ホストで制御を行うことが有効であると考えられる。特に提案方式では、TCP プロトコルに改変を加える必要はなく、またネットワーク構造も変更する必要がない。

以降、2 章では関連研究についてまとめる。3 章では提案方式について説明し、4 章でシミュレーションを用いた性能評価結果を示す。そして、5 章でエンドホスト資源管理方式との統合について考察を示し、最後に 6 章で本稿のまとめと今後の課題を示す。

2. 関 連 研 究

ボトルネックとなるアクセスリンク資源をユーザホスト (受信側ホスト) によって管理する手法は、これまでにも [7,8] などで提案されている。[7] においては、ラストホップルータ (受信側ホストのアクセスリンクに接続されているルータ) のバッファサイズと帯域遅延積を考慮して各 TCP コネクションに割り当てる受信バッファサイズを調整し、インタラクティブ型ネットワークアプリケーションの応答時間の向上と輻輳によって発生するパケット廃棄の減少、およびバルクデータ転送のスループットを高く維持することを目的としている。しかしながら、long-lived コネクションと short-lived コネクションが混在する場合に、long-lived コネクションに割り当てられる受信バッファを 1 パケット分に制限し、short-lived コネクションに多くの受信バッファを割り当てるために、long-lived コネクションのスループットが大きく低下する。また、アクセスリンク帯域に加えて、アクセスリンクで利用可能なバッファサイズを知る必要があり、ユーザに対する負担が大きいという問題がある。[8] においては、受信側ホストでのパケットの受信レートがアクセスリンクでの利用可能帯域に等しいと考え、各 TCP コネクションに設定された優先度、最低転送レート、重みの 3 つのパラメータに従ってアクセスリンク帯域を共有する手法が提案されている。この方式では、主にストリーミングなどのサービスを提供している long-lived コネクションを対象としており、short-lived コネクションに対する考察がなされていない。そのため、仮に short-lived コネクションであってもそのアプリケーションの優先度が大きければ、転送速度が小さいスロースタートフェーズにおいても、他の TCP コネクションに割り当てられていた帯域を減少させるため、アクセスリンクの利用率が低下すると考えられる。

また、文献 [3,9] では、short-lived コネクションと long-lived コネクションが混在する状況における、short-lived コネクションの性能向上に関する研究が行われている。これらの研究は、エンドホストではなくネットワーク内のルータで制御を行うことにより、short-lived コネクションの性能を向上させるための手法を提案している。しかしながら、エッジルータとコアルータの両方が協力する必要があり、導入面での大きな問題がある。さらに、これらの提案方式ではルータバッファの管理機構として RED [10] や CBQ [11] を用いるが、現在のネットワーク上に存在する多くのルータではこれらの機能を利用できないという問題もある。

そこで本稿では、これらの手法の持つ問題点を解決するアクセスリンク資源管理方式として、アクセスリンクの利用率を下げることなく short-lived コネクションを優先的に処理し、ネットワークアプリケーションの特徴を反映させた QoS を考慮して long-lived コネクション間でアクセスリンク帯域を共有する新たな方式を提案する。

3. 提 案 方 式

提案方式における処理は、受信側ホストで観測するネットワーク状況に応じて全ての TCP コネクションに割り当てる受信バッファの総量を仮想的に調整する部分、および各 TCP コネクションへ割り当てる受信バッファサイズを決定する部分に分けることができる。ここで、受信バッファの総量を仮想的に調整するとは、実際の受信バッファサイズではなく、ACK パケットを通して送信側へ伝えられる受信バッファの空き容量、つまり広告ウィンドウサイズを調整することを意味する。

3.1 受信バッファの総量の調整

1 章で述べたように、受信バッファの総量を仮想的に調整するのは、アクセスリンクへのパケットの到着レートを抑え、アクセスリンクでの輻輳を回避するためである。しかし、ネットワーク状況は動的に変化するため、アクセスリンクでの輻輳を回避するためには動的に受信バッファの総量を調整する必要がある。そこで、提案方式においては受信側ホスト上で処理する全ての TCP コネクションの RTT を定期的に観測することによってアクセスリンクの輻輳状態を推測し、下記のように受信バッファの総量を調整する。

- 全ての TCP コネクションの RTT が変化していない場合、アクセスリンク資源にはまだ余裕があると判断し、受信バッファの総量を増加させる。
- 半数以上の TCP コネクションの RTT が増加した場合、受信バッファの総量が多すぎたためにアクセスリンク資源を使いすぎていると判断し、受信バッファの総量を減少させる。
- 上記以外の場合、アクセスリンクで輻輳が発生しておらず、かつアクセスリンク資源が有効に活用されていると判断し、受信バッファの総量は変化させない。

システムが十分なメモリ容量を持ち、受信バッファとして多くのメモリを利用することができる場合にも、受信バッファの総量がここで決定される値を超えないようにすることは重要である。なぜなら、受信バッファの総量を大きく設定しすぎると、各 TCP コネクションの転送パケット量が不必要に上昇するため、アクセスリンクへのパケット到着レートがアクセスリンク帯域を上回り、パケット廃棄を引き起こすためである。

3.2 各コネクションへ割り当てる受信バッファサイズの決定

各 TCP コネクションに割り当てる受信バッファサイズを決定するために、まず各コネクションを short-lived であるか long-lived であるかを判断する必要がある。これは、short-lived コネクションに対しては、そのパケットを優先的に処理し、パケットがアクセスリンクを通って受信側ホストまで到着する割合を向上させ、long-lived コネクションに対しては、ボトルネックとなるアクセスリンク資源をネットワークアプリケーションの特徴を反映させた QoS を考慮して割り当てる必要があるためである。しかし、TCP ではコネクション確立時に転送されるデータサイズを知ることができないため、short-lived か long-lived かを判断することはできない。そこで、提案方式においては受信バッファサイズに関する閾値を設定し、割り当てられた受信バッファサイズがこの閾値を超えたかどうかによって判断する。この判断手法では、確立直後のコネクションは全て同じように short-lived と判断されるため、本稿ではこれらの状態を short-lived、long-lived ではなく initial state、persistent state と表す。閾値は、全ての TCP コネクションを persistent state と仮定したときに割り当てられる受信バッファサイズに設定する。

その後、各 TCP コネクションへ割り当てる受信バッファサイズを以下のように決定する。提案方式においては、initial state コネクションを優先的に処理するためにまず initial state コネクションへ割り当てる受信バッファサイズを決定し、その後、persistent state コネクションへ割り当てる受信バッファサイズを決定する。

(1) initial state コネクションへの割り当て

initial state コネクションに対しては、そのパケットがアクセスリンクを通って受信側ホストまで到着する割合を向上させる必要がある。そこで、initial state コネクションを優先しながらも必要以上に persistent state コネクションのスループットを奪わないようにスロースタートフェーズ中のウィンドウサイズの増加量を考慮し、以下のように受信バッファサイズを決定する。

(1-a) 受信バッファの総量が十分にある場合

この場合は、各 initial state コネクションが要求する受信バッファを割り当てることができる。initial state コネクションはスロースタートフェーズにあると考えられるため、スロースタートフェーズにおけるウィンドウサイズの増加アルゴリズムを考慮し、コネクション開始からの経過 RTT 数 (t_i) に応じて受信バッファサイズを決定する。すなわち、この場合における initial state コネクションの受信バッファ要求量は $2 \cdot 2^{t_i}$ となる。

(1-b) 受信バッファの総量が十分にない場合

この場合は、上述のように各 TCP コネクションが要求する量を割り当てることができない。そこで、各 initial state コネクションの要求量と閾値の差の大きさに従って受信バッファの総量を比例配分する。これは確立直後のコネクションほど優先して、そのパケットが受信側ホストへ到着する割合を向上させる必要があるためである。

以上より、3.1 節で決定した各 TCP コネクションへ割り当てる受信バッファの総量を B、initial state コネクションの本数を N_{is}、initial state と persistent state を判断する閾値を $threshold_i$ とすると、initial state コネクションへ割り当てる受信バッファサイズ ($target_i$) は以下のように表される。

$$target_i = \begin{cases} 2 \cdot 2^{t_i} & \text{(1-a)} \\ B \cdot \dfrac{(threshold_i - 2 \cdot 2^{t_i})}{\sum_j^{N_{is}} (threshold_j - 2 \cdot 2^{t_j})} & \text{(1-b)} \end{cases}$$

(2) persistent state コネクションへの割り当て

persistent state コネクションに対しては、ボトルネックとなるアクセスリンク資源を有効に活用するために QoS を考慮する必要がある。そこで、ネットワークアプリケーションの特徴を考慮した優先度に従って、以下のように受信バッファサイズを決定する。

(2-a) 受信バッファの総量が十分にある場合

この場合、全ての initial state コネクションの総要求量以上に受信バッファを割り当てることができるため、initial state コネクションに受信バッファを割り当てた余りを各 TCP コネクションの優先度および RTT に応じて割り当てる。

(2-b) 受信バッファの総量が十分にない場合

この場合は、persistent state コネクションに対して割り当てる受信バッファサイズが不足する。しかし、Silly window syndrome [12] を回避するために、各 TCP コネクションの最大セグメントサイズ (1 mss_i) 分の受信バッファサイズを最低限割り当てる必要がある。

以上より、initial state コネクションに割り当てられた受信バッファの総量を T_{is}、persisntent state コネクションの本数を N_{ps}、各 TCP コネクションの RTT を rtt_i、各 TCP コネクションごとに設定された優先度を p_i とすると、persistent state コネクションへ割り当てる受信バッファサイズ ($target_i$) は以下のように表される。

$$target_i = \begin{cases} (B - T_{is}) \cdot \dfrac{p_i \cdot rtt_i}{\sum_j^{N_{ps}} (p_j \cdot rtt_j)} & \text{(2-a)} \\ 1\ mss_i & \text{(2-b)} \end{cases}$$

3.3 ボトルネックの特定

提案方式では、受信側ホスト上で処理する全ての TCP コネクションの RTT の変化からアクセスリンクの輻輳状態を推測する。そのため、アクセスリンク以外のリンクがボトルネックになり RTT が増加する場合に、誤判断を引き起こす可能性がある。しかし、受信側ホスト上で処理する多くの TCP コネクションの RTT が増加した場合、これらの TCP コネクションは同じリンク上で発生した輻輳の影響を受けている可能性が高く、この場合に輻輳が発生しているのはアクセスリンクであると考えることができる。したがって、提案方式においては受信側ホスト上で処理する半数以上の TCP コネクションの RTT が増加した場合、アクセスリンク帯域がボトルネックであると判断する。一方、受信側ホスト上で処理する一部分の TCP コネクションの RTT のみ増加した場合、それらの TCP コネクションだけが通るリンク上で発生した輻輳の影響を受けている可能性が高く、アクセスリンク以外の箇所で輻輳が発生していると考えることができる。したがって、提案方式においては、受信側ホスト上で処理する、一部分の TCP コネクションの RTT が増加した場合、アクセスリンク以外のリンクがボトルネックであると判断する。またこれ以外にも、1 章で述べたようにアクセスリンク資源ではなくエンドホスト資源がボトルネックとなる場合もある。このような場合に対しては、我々が提案したエンドホストにおける TCP コネクション資源の管理方式 [1] を組み合わせて利用することが有効であると考えられる。詳細については 5 章で説明する。

4. シミュレーションによる性能評価

本章では、ns-2 [13] を用いたシミュレーションによって、標準の TCP を用いた場合と文献 [7] および [8] の手法を用いた場合との性能比較を行い、提案方式の有効性を評価する。本章では、文献 [7] において提案された手法を Spring、文献 [8] において提案された手法を Mehra とする。シミュレーションで用いたネットワークトポロジを図 2 に示す。このシミュレーショントポロジにおいて、送信側ホスト A-D と受信側ホスト間の伝播遅延時間はそれぞれ A : 35 msec、B : 45 msec、C : 55 msec、D : 10 msec である。以下のシミュレーションでは、アクセスリンク帯域 (4Mbps) がボトルネックとなるように、送信側ホスト A、B、C から無限長サイズのバルクデータ転送を行い (long-lived コネクション) 、アクセスリンク帯域を使い切る。そして、同時に送信側ホスト D から 100 本の short-lived コネクション (転送データ量 : 30KB、コネクションの最大生存期間 10 秒) をシミュレーション開始後 100 秒の時点から 1 本ずつランダムに確立する。提案方式における観測期間は 5 sec とし、文献 [7, 8] の提案手法との性能比較を行うためにこれらの手法におけるパラメータとして表 1 の値を用いる。これらのパラメータは文献中で推奨されている値であり、シミュレーショントポロジの違いがパラメータ選択へ与える影響はほとんどない。

図 3 に各 short-lived コネクションにおける、コネクション確立時間とデータ転送完了時間に関する累積度数分布、シミュレーション開始から 500 秒間の全てのコネクションの合計スループット、ラストホップルータ (受信側ホストのアクセスリンクに接続されているルータ) のバッファの平均キュー長を示す。ここでスループットとは、受信側ホストが 1 秒間あたりに受信した総データ量を表す。図中では、提案方式を用いた場合のシミュレーション結果を proposed 、文献 [7] の手法を用いた場合を Spring、文献 [8] の手法を用いた場合

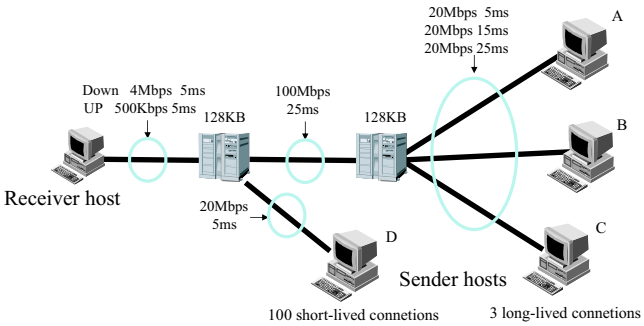


図 2　シミュレーショントポロジ

表 1　文献 [7,8] の手法のパラメータ

Spring (文献 [7])		Mehra (文献 [8])	
アクセスリンク帯域	4Mbps	優先度	0
キュー長 (輻輳回避)	64KB	最低転送レート	0
キュー長 (遅延減少)	5KB	重み	0
閾値 (interactive-short)	2KB		
閾値 (short-long)	8KB		

を Mehra、何も制御を行わない標準の TCP を用いた場合を traditional と示している。図 3(a) および図 3(b) より、何も制御を行わない標準の TCP を用いた場合、short-lived コネクションの確立時間とデータ転送完了時間が長いことがわかる。これは、標準の TCP ではアクセスリンク資源を正確に推測することができず、アクセスリンクで発生した輻輳によって、ラストホップルータのバッファにおいてパケット廃棄が発生しやすいためである。また、図 3(c) より全体のスループットは十分高いことがわかる。しかし、この高いスループットはほとんど long-lived コネクションのパケットによって占められており、short-lived コネクションのスループットは低く抑えられている。

図 3(a) および図 3(b) の結果より、Mehra を用いた場合が、short-lived コネクションの確立時間とデータ転送完了時間が最も小さいことがわかる。しかし、図 3(c) より、この場合のスループットが最も低いことがわかる。Mehra では short-lived コネクションと long-lived コネクションを区別せず同じ帯域を割り当てようとし、アクセスリンク帯域を 3 本の long-lived コネクションと 1 本の short-lived コネクションで等しく共有する。そのため、short-lived コネクションの転送終了後もすぐに short-lived コネクションに割り当てていた帯域をすぐに long-lived コネクションが利用できず、スループットが低下していると考えられる。図 3(d) より、Merha を用いた場合では、ラストホップルータの平均キュー長が最も小さいことがわかるが、これは頻繁にキューが空になり、アクセスリンクの利用率が低下しているためである。また、図 3(d) より、Spring を用いた場合、ラストホップルータの平均キュー長が比較的大きいことがわかる。これは、Spring では、ラストホップルータのバッファの 1/2 を利用するように各コネクションへ受信バッファを割り当てるためである。そのため、図 3(a) および図 3(b) より short-lived コネクションにおけるデータ転送時間が大きくなっていることがわかる。一方、提案方式では高いスループットを維持しながらも、ラストホップルータでの平均キュー長は小さく、short-lived コネクションのデータ転送時間も大きく減少していることがわかる。

long-lived コネクションのスループットの変化を図 4 に示す。図中において、送信側ホスト A とのコネクションにおけるスループットを flow A、送信側ホスト B とのコネクションにおけるスループットを flow B、送信側ホスト C とのコネクションにおけるスループットを flow C と表す。また、best は最も公平かつ有効にアクセスリンク帯域が共有された場合のスループット値である。図 4 より、何も制御を行っていない場合と比較し、提案方式および Spring、Mehra を用いると、long-lived コネクション間のスループットの公平性が改善されていることがわかる。しかし、Mehra を用いた場合、図 4(c) よりスループットの変動が大きいことがわかる。これは、今回のシミュレーションでは Mehra がパラメータとして持つ優先度、重み、最低転送レートを設定していないため、全てのコネクション間で等しく帯域を共有するように調整を繰り返すためであると考えられる。これに対し、提案方式や Spring を用いた場合では、図 4(a)、図 4(b) より各コネクションごとのスループットの変動が小さく、Mehra を用いた場合よりもアクセスリンク資源を有効に利用していることがわかる。これは、アクセスリンク資源に注目した制御を行っているため、アクセスリンク資源を十分に活用しながらもアクセスリンクにおいて輻輳が発生していないことを示している。しかし、Spring においては、あらかじめアクセスリンク帯域およびバッファサイズを知る必要があるため、特に複数ユーザでアクセスリンクを共有しているネットワークなどにおいては適用できない。一方、提案方式では動的にネットワーク状況を判断するため、アクセスリンク資源が既知である必要はなく、動的な資源量の変化に対しても追随することが可能である。

5. エンドホスト資源管理方式との統合

我々はこれまでに、主に Web / Web プロキシサーバなど、多数のコネクションが頻繁に張られるようなエンドホストを対象に、エンドホストにおける TCP コネクション 資源の管理方式を提案した [1]。そこで、本稿で提案したアクセスリンク資源管理方式と、 [1] の手法を統合することによって、状況によってボトルネックとなる資源が変化するようなエンドホストにも適用することができると考えられる。たとえば、P2P ネットワークにおいては、隣接ノード数が非常に大きいノードが存在することが知られており [15]、ノードが接続されているアクセスリンク帯域が十分大きくボトルネックにならない場合においても、エンドホスト資源が不足する可能性がある。また P2P ネットワークにおいてはそのネットワーク構成が動的に変動するため、ボトルネックが変化することが考えられる。

そこで本章では、このようにボトルネックとなる資源の動的な変化に対応するために、エンドホスト資源管理方式とアクセスリンク資源管理方式を統合した手法を提案する。統合した手法においてはボトルネックとなる資源を動的に判定し、その結果に応じた資源管理を行う。なお、ここではエンドホスト資源として受信バッファの総量を用いる。

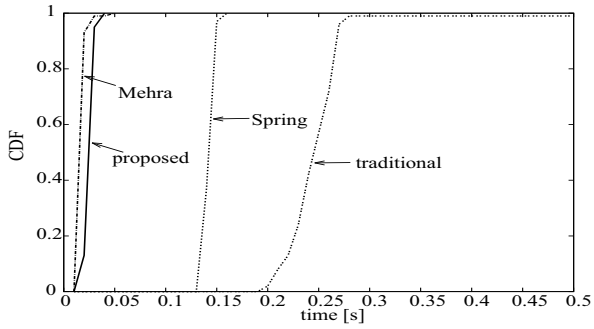
5.1 統合方式における資源管理方法

まず、本稿で提案したアクセスリンク資源管理方式によって、必要と考えられる受信バッファの総量を求める。そして、この受信バッファの総量とエンドホストにおいて割り当てることができる受信バッファの総量を比較し、エンドホストにおいて割り当てることができる受信バッファの総量が不足した場合は、アクセスリンク管理方式によって求めた各コネクションへの受信バッファの割り当てからこの不足分を減少させる。その後、 [1] において提案しているエンドホスト資源管理方式に従って各 TCP コネクションの受信バッファの使用量を監視し、ネットワーク状況から必要と思われる以上に受信バッファが割り当てられていると判断したコネクションがあれば、このコネクションに割り当てた受信バッファを減らし、アクセスリンク資源管理方式によって求めた各コネクションへの受信バッファの割り当てを超えないように、この減少分を他のコネクションに配分する。これは、アクセスリンクがボトルネックであることを前提としている本稿の提案手法によって割り当てられた受信バッファが使い切られないことは、そのコネクションにおいてはアクセスリンク以外のリンクがボトルネックになっていることを意味しているためである。この手法により、エンドホスト資源を有効に活用しつつ、アクセスリンク資源も有効に利用することができると考えられる。

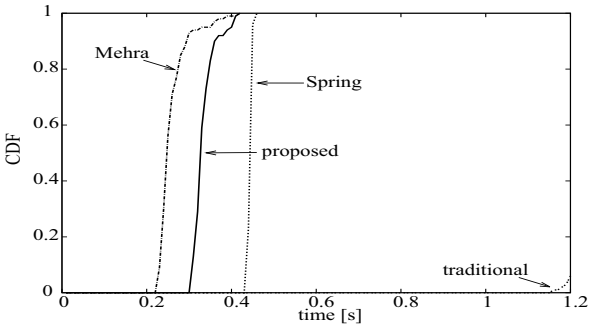
5.2 シミュレーションによる有効性の確認

本節では ns-2 [13] を用いたシミュレーションによって、統合方式の有効性を確かめる。シミュレーショントポロジを図 5(a) に示す。シミュレーションでは、100 Mbps のリンクを共有する送信側ホスト A、B から無限長サイズのバルクデータ転送を行う。そして、リンク C においてバックグラウンドトラヒックとして 3.5 Mbps の UDP トラヒックをシミュレーション開始から 100 秒経過後から 500 秒経過後まで発生させる。これにより、この期間のホスト A にとってのボトルネックはリンク C となる。

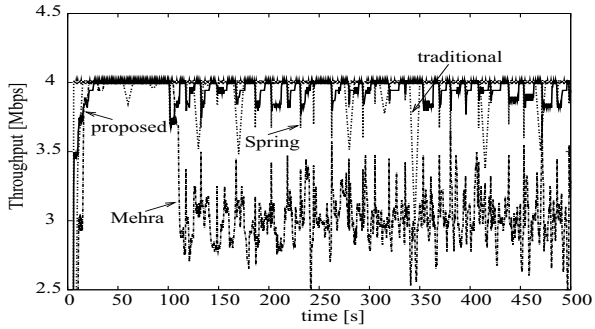
図 5(b) に各 TCP コネクションにおけるスループットの変化を示す。図中において送信側ホスト A とのコネクションにおけるスループットを flow A、 送信側ホスト B とのコネ



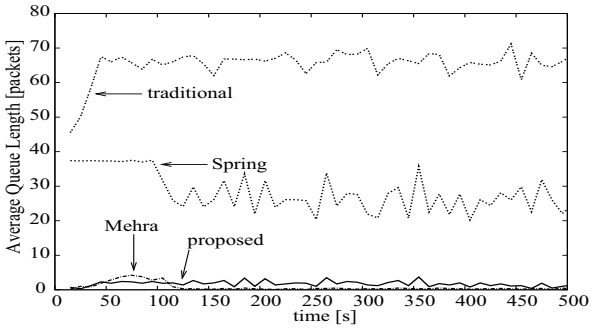
(a) short-lived コネクション確立時間分布



(b) Web ドキュメントの転送時間分布



(c) 平均スループット



(d) 平均キュー長

図 3　シミュレーション結果 (1)

クションにおけるスループットを flow B と表す。この結果より、バックグラウンドトラヒックが発生している 100 秒から 500 秒の区間において、この影響を受ける flow A のスループットがおよそ 500 Kbps に減少しているが、一方で flow B のスループットが増加していることがわかる。これは、バックグラウンドトラヒックによって flow A のスループットが減少した結果、それまでアクセスリンク資源管理方式によって割り当てられていた受信バッファサイズでは大きすぎるため、その余剰分がエンドホスト資源管理方式によって flow B へ再配分され、アクセスリンク資源を有効に活用できているためである。

6. おわりに

本稿では、受信側ホストによるアクセスリンク資源管理手法を提案した。提案方式では、アクセスリンクでの輻輳を回避するために全ての TCP コネクションに割り当てられる受信バッファの総量を仮想的に調整し、short-lived コネクションに対しては、そのパケットがユーザホストまで到着する割合を向上させ、long-lived コネクションに対しては QoS を考慮して受信バッファの割り当てを行う。提案方式の有効性はシミュレーションによって評価した。その結果、提案方式を用いることによって、アクセスリンク資源がボトルネックとなる場合にその資源を有効に活用することができ、long-lived コネクションのスループットを高く維持したまま、short-lived コネクションの確立時間およびデータ転送時間が向上することが明らかになった。また、文献 [7, 8] で提案された手法とのシミュレーションを用いた性能比較を行い、提案方式の優位性を明らかにした。

また、過去にわれわれが提案したエンドホスト資源管理方式と統合させることによって、アクセスリンク資源がボトルネックとなる場合に加えて、エンドホスト資源がボトルネックとなる場合にも適用することができることを明らかにした。この結果、提案方式は、多くの TCP コネクションが確立され、バルクデータ転送が行われる P2P ネットワークにおけるノードのようにアクセスリンク資源とエンドホスト資源の両方の資源が要求されるような場合にも有効であると考えられる。

今後の課題としては本方式を実システム上へ実装し、実ネットワーク上での性能評価を行うことが挙げられる。

謝　辞

本研究の一部は、総務省における研究プロジェクトである戦略的情報通信研究開発推進制度委託研究「ユビキタスインターネットにおける高位レイヤスイッチの研究開発」によっている。ここに記して謝意を表す。

文　献

[1] T. Okamoto, T. Terai, G. Hasegawa, and M. Murata, “A resouorce/connection management scheme for HTTP proxy servers,” in *Proceedings of Second International IFIP-TC6 Networking Conference*, pp. 252–263, May 2002.

[2] J. Padhye, V. Firoiu, D. Towsley, and J. Krusoe, “Modeling TCP throughput: A simple model and its empirical validation,” in *Proceedings of ACM SIGCOMM ’98*, pp. 303–314, Aug. 1998.

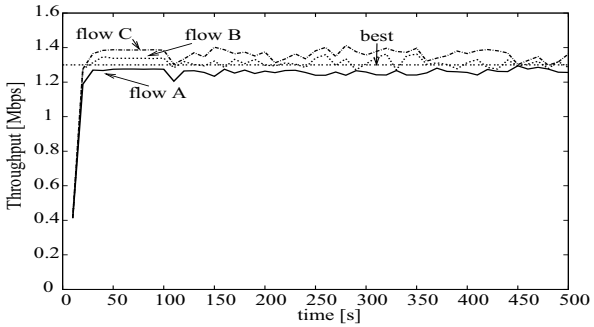
[3] L. Guo and I. Matta, “The war between mice and elephants,” in *Proccedings of the 9th IEEE International Conference on Network Protocols*, no. 2001-005, Nov. 2001.

[4] W3C Recommendations Reduce ’World Wide Wait’, available at `http://www.w3.org/Protocols/NL-PrefNote.html`.

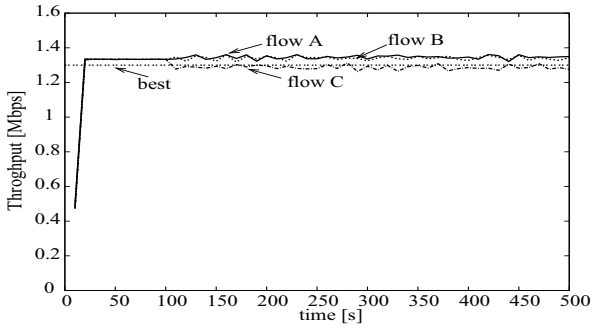
[5] H. Balakrishnan, H. S. Rahul, and S. Seshan, “An integrated congestion management architecture for Internet hosts,” in *SIGCOMM 1999*, pp. 175–187, Sept. 1999.

[6] J. Touch, “TCP control block interdependence,” *Request for Comments (RFC) 2140*, Apr. 1997.

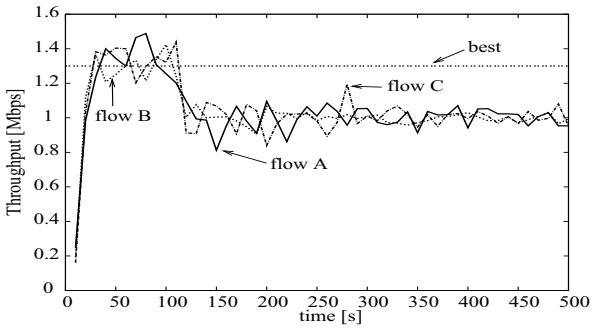
[7] N. T. Spring, M. Chesire, M. Berryman, V. Sahasranaman, T. Anderson, and B. N. Bershad, “Receiver based management of low



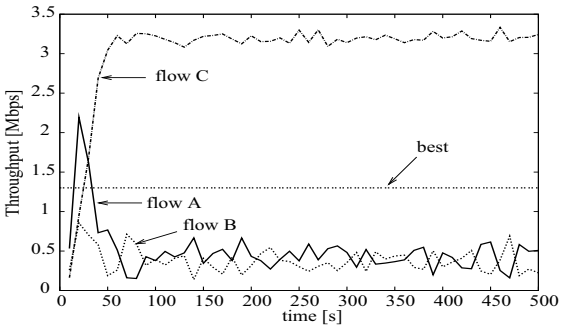
(a) 提案方式



(b) Spring

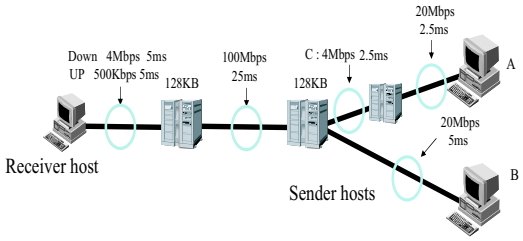


(c) Mehra

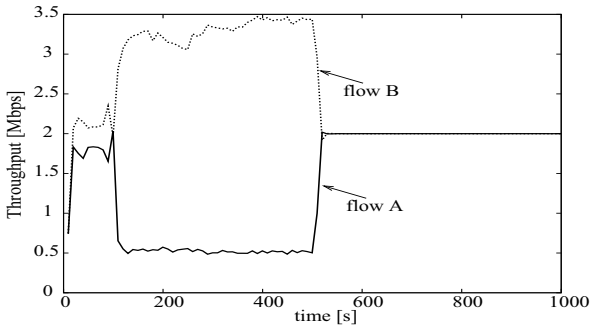


(d) 標準 TCP

図 4　シミュレーション結果 (2)



(a) シミュレーショントポロジ



(b) 平均スループット

図 5　統合方式におけるシミュレーション結果

bandwidth access links," in *Proccedings of IEEE INFOCOM2000*, pp. 245–254, 2000.

[8] P. Mehra, A. Zakhor, and C. D. Vleeschouwer, "Receiver-driven bandwidth sharing for TCP," in *Proceedings of IEEE INFOCOM2003*, Mar. 2003.

[9] M. Hartling, M. Claypool, and R. Kinicki, "Active queue management for Web traffic," Tech. Rep. WPI-CS-TR-02-20, Computer Science Department, Worcester Polytechnic Institute, May 2002.

[10] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, pp. 397–413, Aug. 1993.

[11] S. Floyd and V. Jacobson, "Link-sharing and resource management models for packet networks," *IEEE/ACM Transactions on Networking*, vol. 3, pp. 365–386, Aug. 1995.

[12] D. D. Clark, "window and acknowledgement strategy in TCP," *Request for Comments (RFC) 813*, July 1982.

[13] The VINT Project UCB/LBNL/VINT network simulator - ns (version 2), available at `http://www.isi.edu/nsnam/ns/`.

[14] R. Jain, A. Durresi, and G. Babic, "Throughput fairness index: An explanation," *ATM Forum Contribution: AF/99-0045*, Feb. 1999.

[15] L. A. Adamic, R. M. Lukose, A. R. Puniyani, and B. A. Huberman, "Search in power-law networks," *Physical Review E 64 46135*, 2001.