無線LAN環境におけるTCP不公平性緩和のためのトランスポート層ソリューション

A Transport-layer Solution for Alleviating TCP Unfairness in aWireless LAN Environment

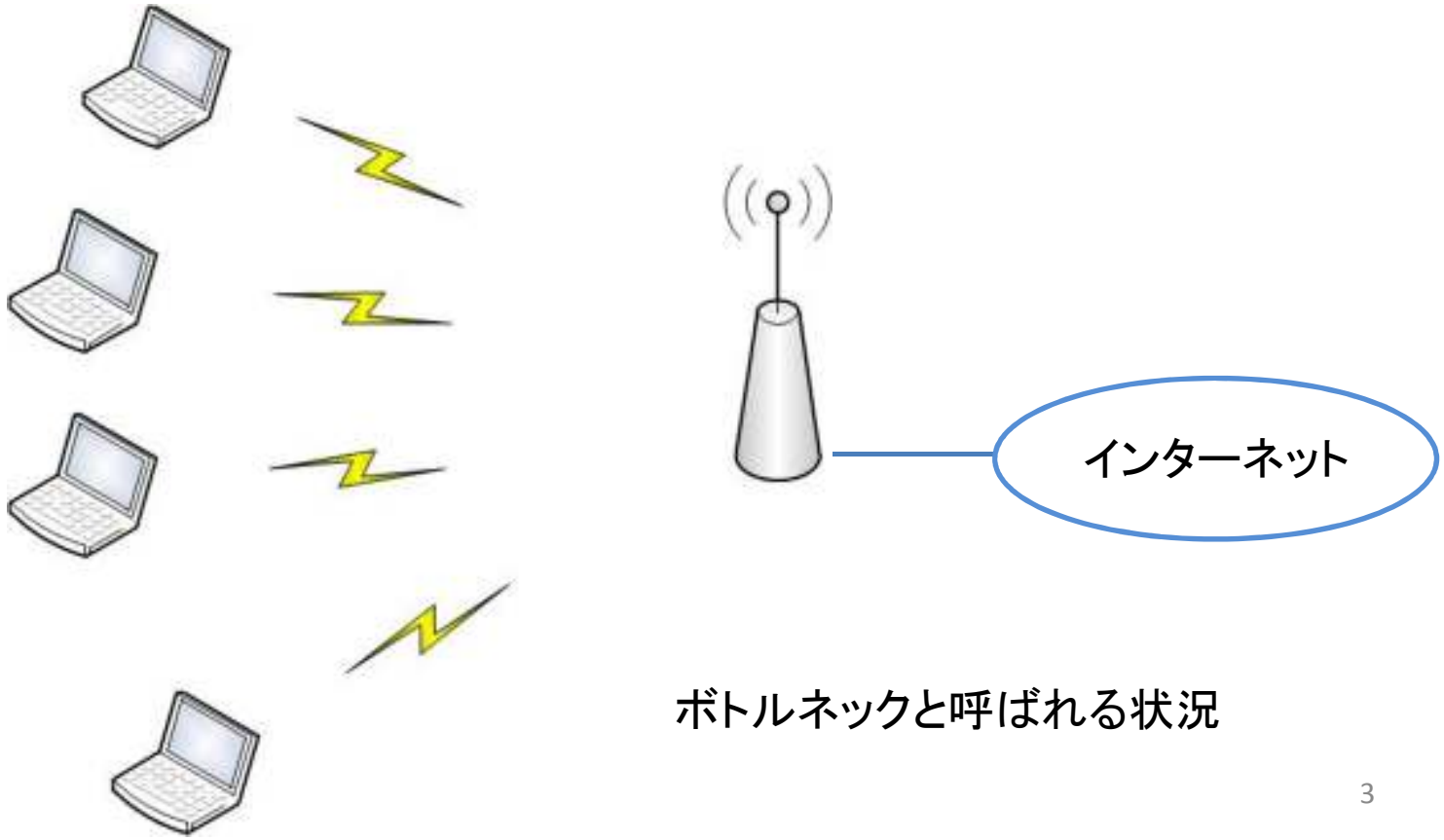
三鴨勇太

文献について

- A Transport-layer Solution for Alleviating TCP Unfairness in aWireless LAN Environment
  - 著者 : Masafumi HASHIMOTO), Nonmember, Go HASEGAWA, and Masayuki MURATA
  - IEICE TRANS. COMMUN., VOL.E94-B, NO.3 MARCH 2011
  - 詳細または正確な知識を求める方は文献を参照願います

環境

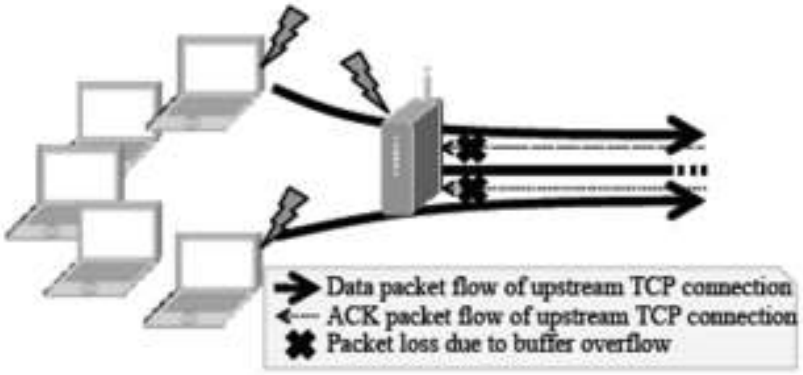
- 1つのアクセスポイント(AP)を複数のクライアント(ステーション)で共有している



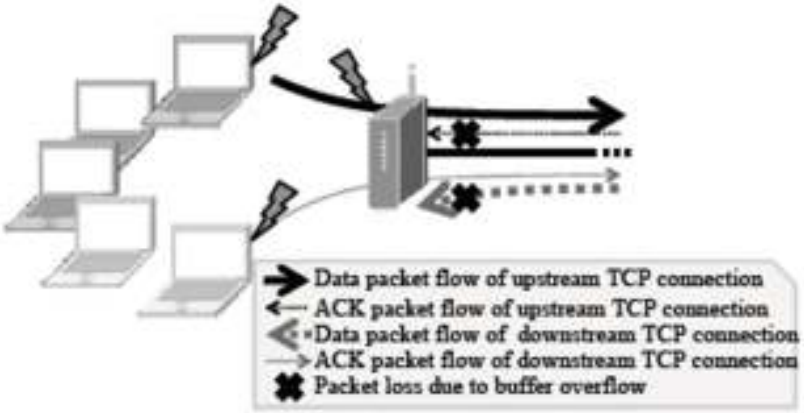
ボトルネックと呼ばれる状況

問題

- TCPフロー間の公平性の問題
  - 上り通信フロー間
  - 上り通信フローと下り通信フロー



(a) Multiple upstream TCP flows exist



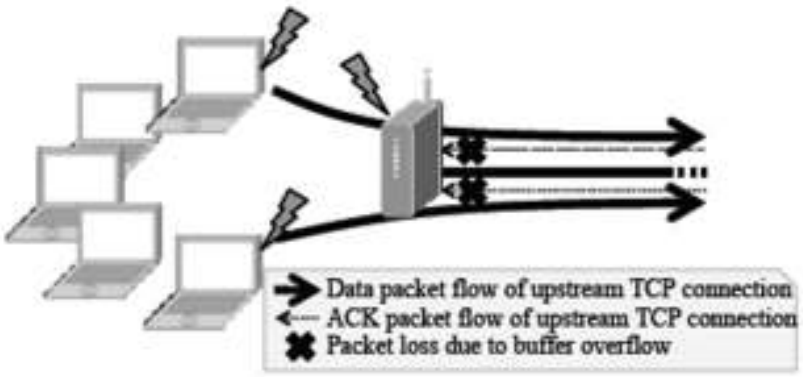
(b) Upstream and downstream TCP flows coexist

背景

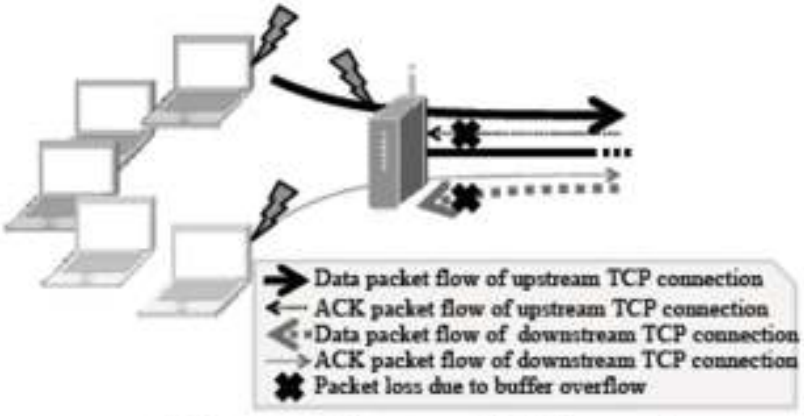
- 無線LANの普及
  - 駅，空港など　公共スペース
- 双方向通信
  - P2Pファイル共有
  - オーディオ/ビデオ会議アプリケーション

考察されている原因

- APのバッファにおいて大量のTCP ACKパケットが破棄されるほどの輻輳が発生している



(a) Multiple upstream TCP flows exist



(b) Upstream and downstream TCP flows coexist

輻輳:ものが1ヵ所に集中新混雑する状態

TCP　輻輳制御について

- 低速回線では低速に，高速回線では高速に
- ネットワークが混雑しているときは速度を落とし，衝突を回避する

TCP アルゴリズム

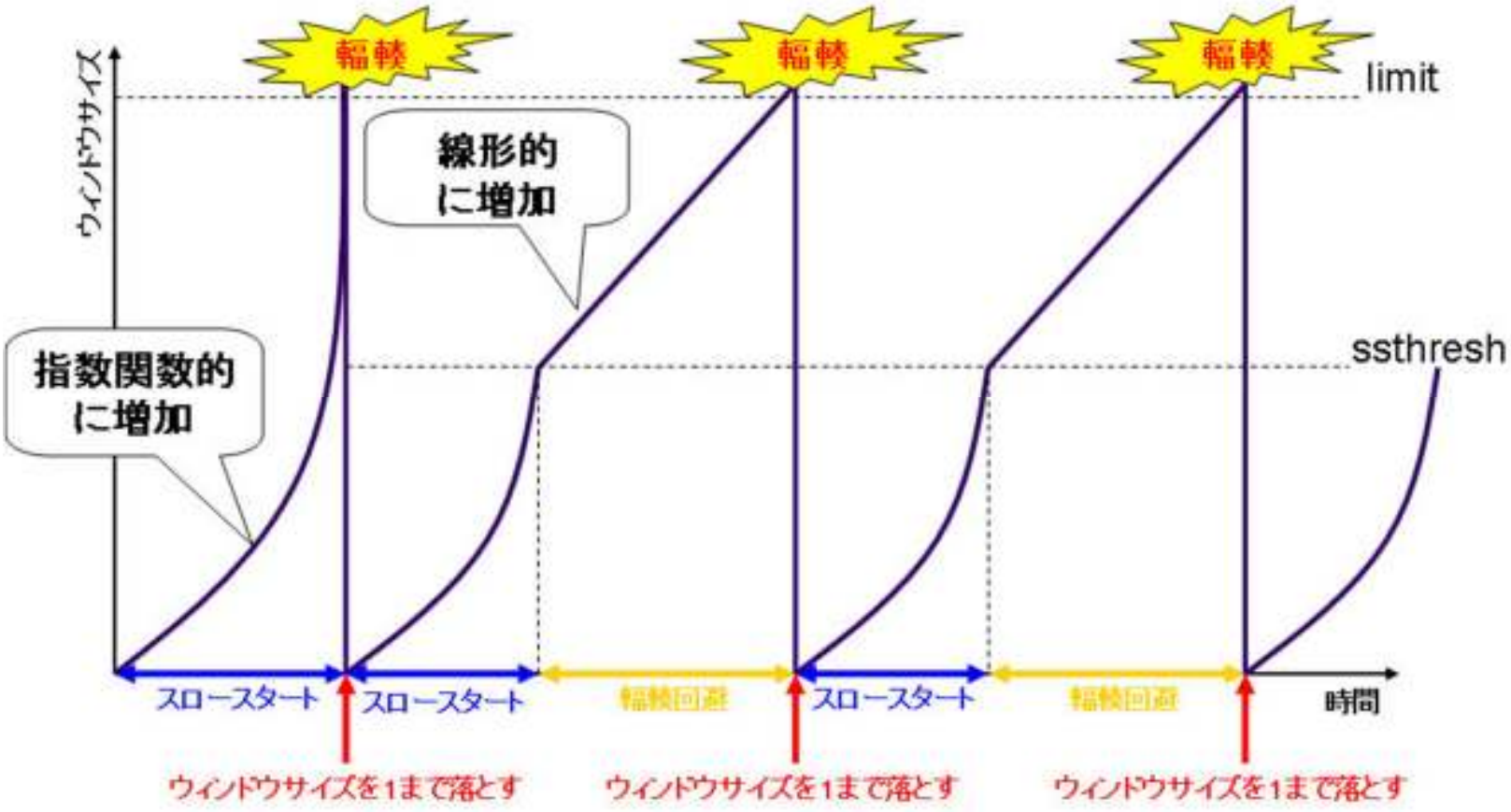
表 2：TCP 輻輳制御アルゴリズムの歴史

88 年頃	Tahoe	スロースタートアルゴリズム、輻輳回避アルゴリズム、Fast Retransmit アルゴリズムの採用
90 年頃	Reno	Fast Recovery アルゴリズムの採用(輻輳の度合が少ない場合、転送速度を大きく落さないのが狙い)
96 年頃	NewReno	Fast Recovery アルゴリズムの修正(パケットの喪失率がやや大きい場合に対するアルゴリズムの不具合の修正)
99 年	RFC2581	Fast Recovery アルゴリズムを採用

TCP詳説　西田 佳史　(株)ソニーコンピュータサイエンス研究所　(1999年12月14日)
http://www.nic.ad.jp/ja/materials/iw/1999/notes/C3.PDF

TCP Tahoe / TCP Reno

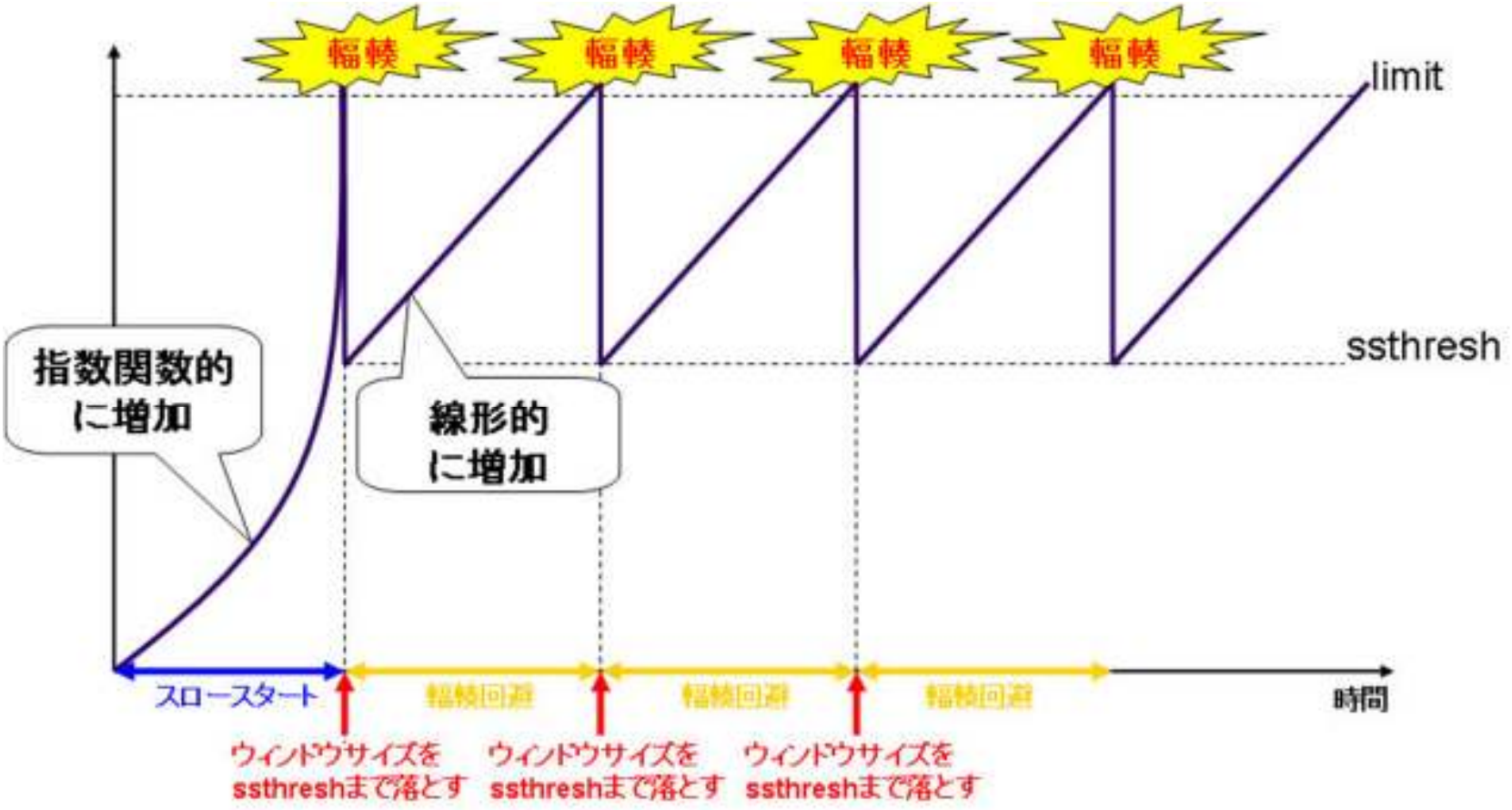
- TCP Tahoe



http://www.net.c.dendai.ac.jp/~yutaro/ より

TCP Tahoe / TCP Reno

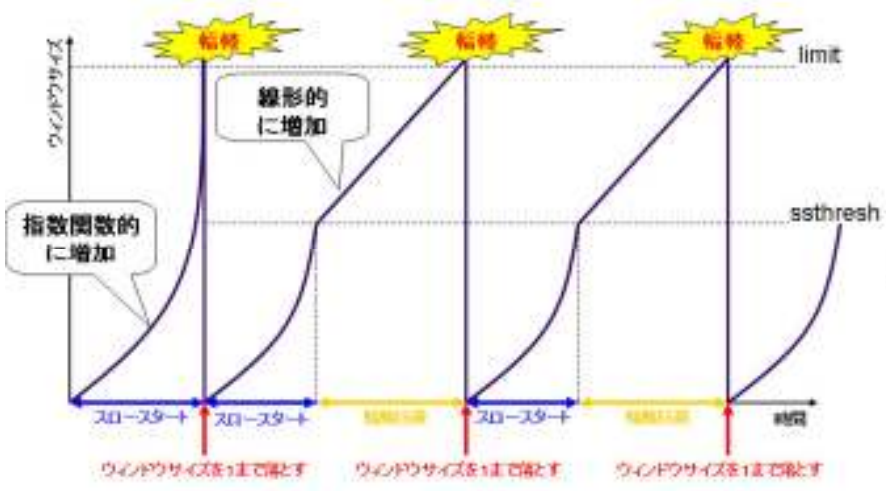
- TCP Reno



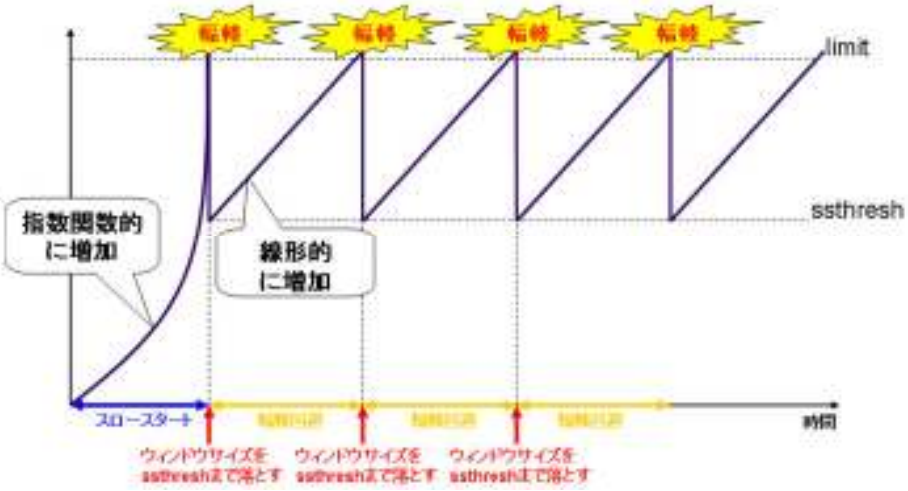
http://www.net.c.dendai.ac.jp/~yutaro/ より

TCP Tahoe / TCP Reno

TCP Tahoe

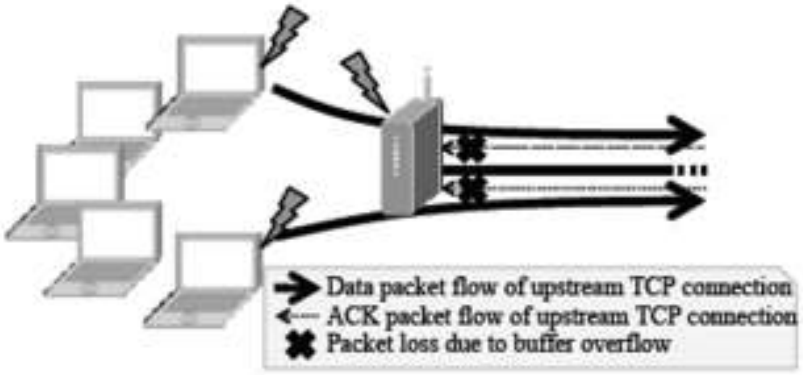


TCP Reno

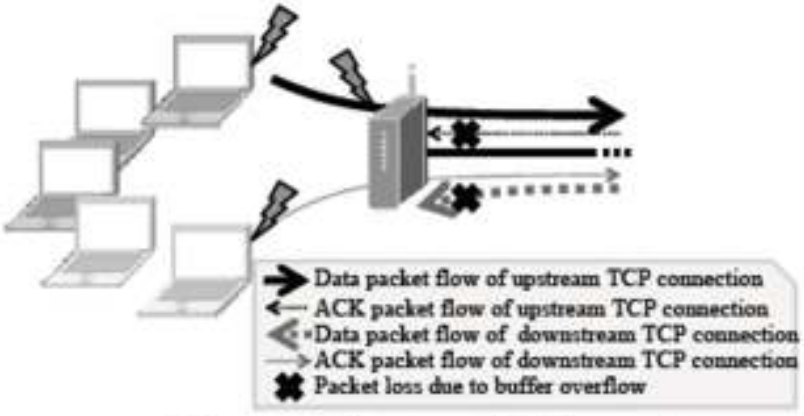


問題

- TCPフロー間の公平性の問題
  - 上り通信フロー間
  - 上り通信フローと下り通信フロー



(a) Multiple upstream TCP flows exist



(b) Upstream and downstream TCP flows coexist

過去の提案

- APのバッファを分割し，ACKバッファを設ける
- 受信ウィンドウサイズを書き換える
- APにACKパケットをフィルタリングさせる
  - これらはMACプロトコルまたはキュー管理メカニズムを変更しなければならない
  - TCPで公平性が向上するが，UDPで不公平性が発生することがある

MACプロトコルについて

- データリンク層で利用するプロトコル
- ハードウェアレベルで実装されているため変更にはコストがかかる

<table>
<tr><td rowspan="2">データリンク層</td><td>LLC副層(論理リンク制御層)</td></tr>
<tr><td>MAC副層(メディアアクセス制御層)</td></tr>
</table>

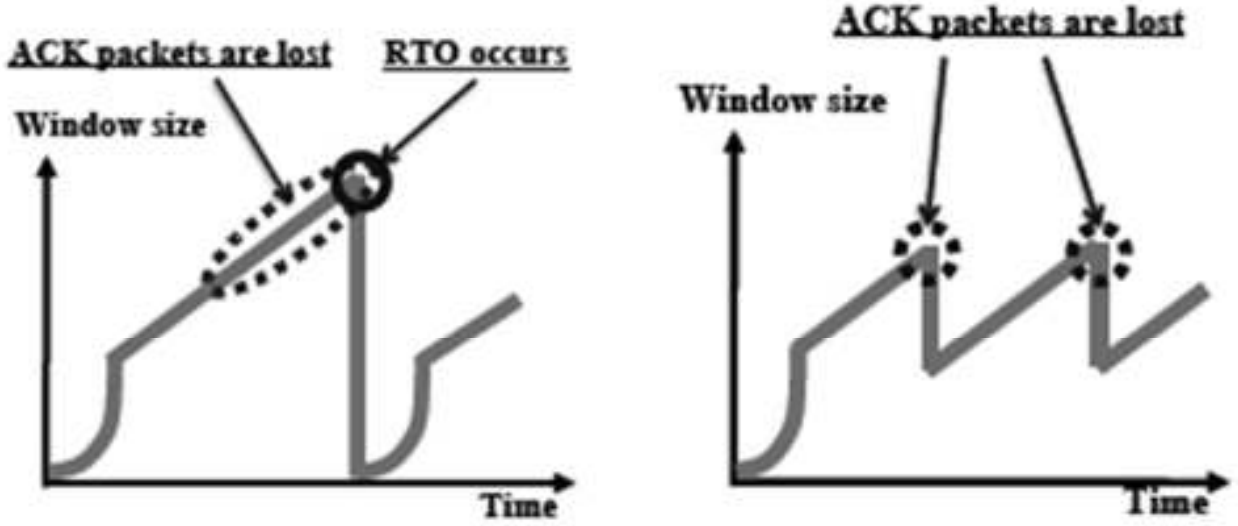
http://d.hatena.ne.jp/n_euler666/20070815/1187103988 より

提案手法について(1)

- トランスポート層でのアプローチ
  - MAC層プロトコル提案の問題点
  - 不公平性の原因は主にトランスポート層プロトコル
  - トランスポート層プロトコルはソフトウェアで実装されているため，変更が容易である
- APのトラフィックの混雑の目安としてTCP ACKパケットの損失を検出
  - TCP輻輳制御メカニズムへの小さな変更で可能

提案手法について(2)

- ACKパケット損失のためのTCP輻輳制御
  - (a) APバッファでACKパケット破棄が起こっているときの動作(RTOにより輻輳制御)
  - (b) 提案手法を適用したときの動作



(a) Without modification (b) Proposed method

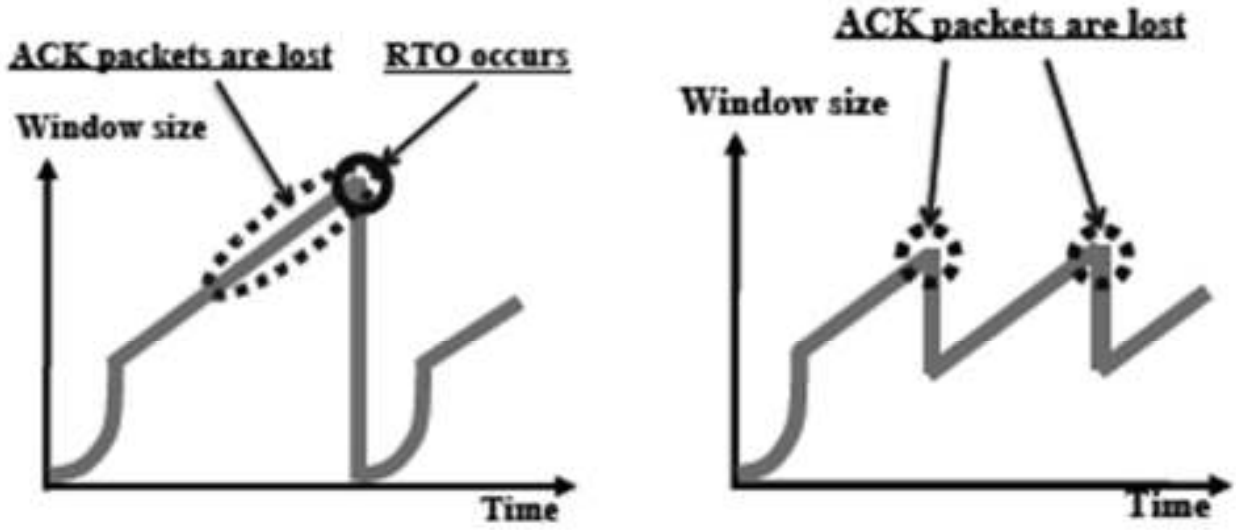
Fig. 2 Behaviors of TCP Reno with and without the proposed method

提案手法について(3)

- メカニズムの変更が必要
  - 従来:データパケットが損失した場合のみ輻輳制御を行う
  - 提案:ACKパケットが損失した場合にも輻輳制御を行いウィンドウサイズを小さくする

提案手法について(4)

- ACKパケットの損失を検出した場合にも輻輳制御を適用している



(a) Without modification (b) Proposed method

Fig. 2 Behaviors of TCP Reno with and without the proposed method

提案手法(5)

- 具体的にはウィンドウ内のACKパケットの損失数が所定の閾値(thresh ack losses)を超えた場合に輻輳制御を適用する
- TCP送信者が受信されたACKのシーケンス番号を監視し，ACKパケット損失を検出する
- RTT(Round Trip Time)のACKパケット損失数が閾値を超えたとき，TCP送信者が輻輳ウィンドウサイズを半分に，スロースタート閾値は輻輳ウィンドウサイズの半分に設定される

比較検証（1）

環境

- ns-2によるIEEE802.11aのWLANシミュレーション環境
- 複数のクライアントが1つのAPを共有
- APは100Mbpsの有線リンクを介して有線ノードへ接続
  - 一方向100msの伝送遅延
- クライアントはAPから4mの位置に設置
- APのバッファサイズ100パケット

比較検証（2）

- 上り通信フロー間の公平性の検証

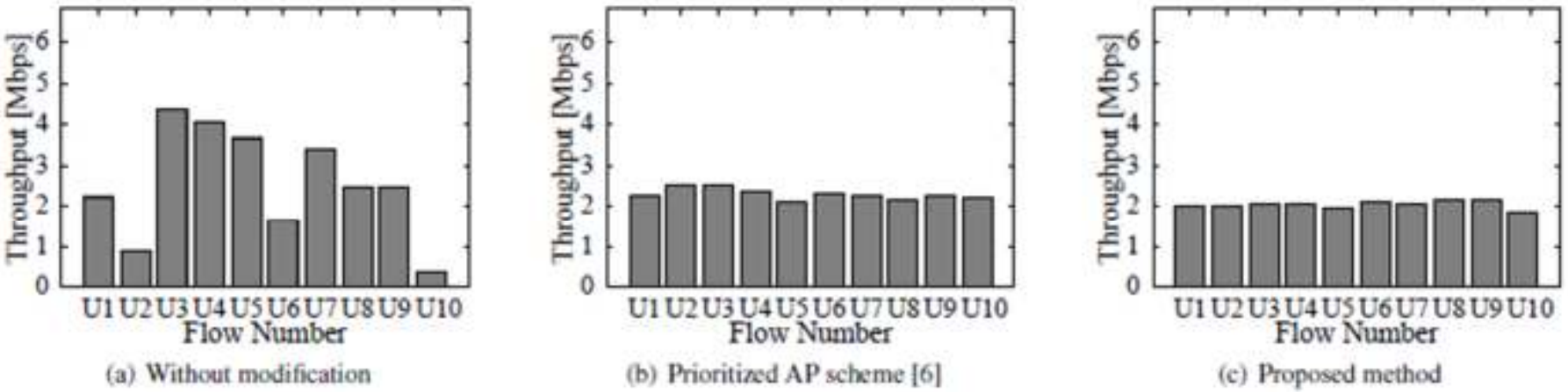


Fig. 4 Average throughput of each flow when ten upstream flows exist

(a)既存方式 (b)優先AP通信方式 (c)提案方式

比較検証（3）

- 上り通信フローと下り通信フローとの間の公平性の検証

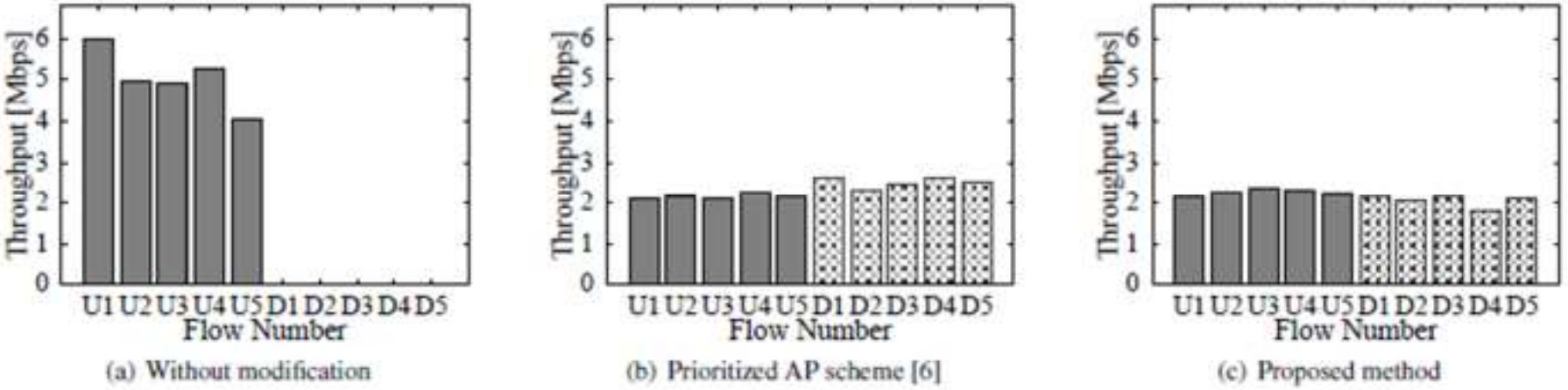


Fig. 5 Average throughput of each flow when five upstream flows and five downstream flows coexist

(a)既存方式 (b)優先AP通信方式 (c)提案方式

比較検証（4）

優先AP通信方式との比較

- 上り通信フローのみ
- 5つのクライアントをAPから1mの位置，残り5つを10mの位置に設置

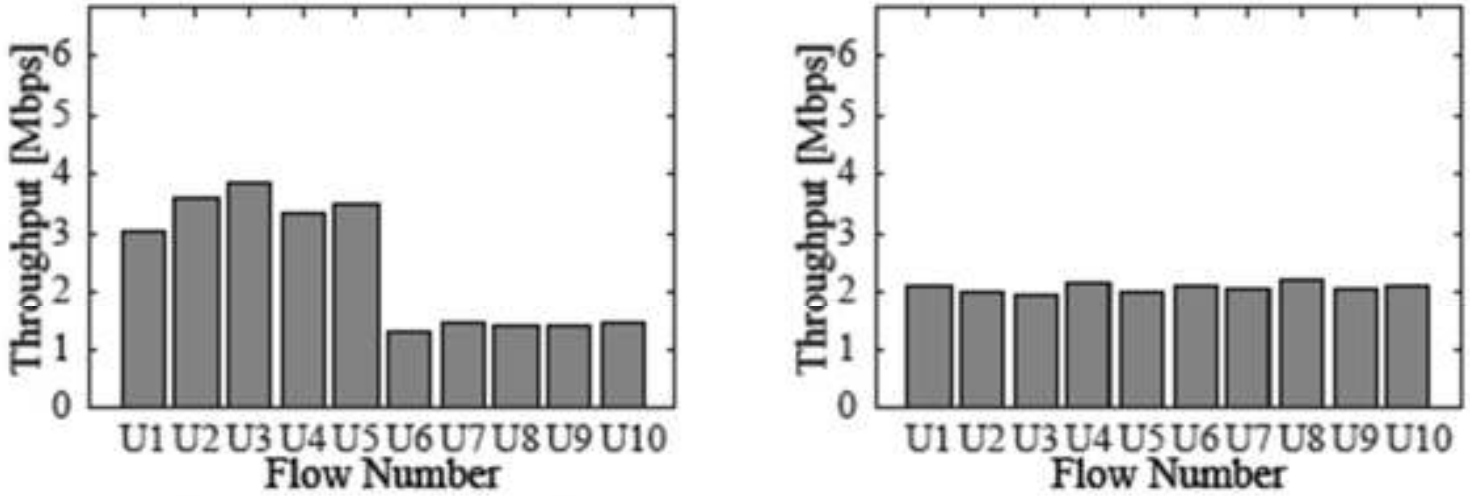


Fig. 6 Average throughput of each flow when ten upstream flows exist and stations are relocated

今後の課題について

- 提案手法はTCPの種類を考慮する必要はなく，任意のTCPに変更を行うことで利用できる

- 従来のTCPと提案手法を適用したTCPが共存する場合においての効果がみられない

指標の提案

- 公平性と使用率との間のトレードオフを評価するための指標を提案

- これまでの提案（指標）では無線チャンネル上のすべてのフローが同じスループットを実現するよう定義されていて，ネットワーク全体のスループットへの影響は考慮されていない

提案指標

- 過去の提案としてJain氏の提案している指標[1]との比較がされている

Table 1 Comparison between Jain's fairness index and the proposed index (C = 30 Mbps)

Case	Throughput distribution [Mbps]	Total [Mbps]	Jain's index	Proposed index
1	{3, 3, 3, 3, 3, 3, 3, 3, 3, 3}	30	1.00	1.00
2	{2, 2, 2, 2, 2, 2, 2, 2, 2, 2}	20	1.00	0.90
3	{1, 1, 1, 1, 1, 1, 1, 1, 1, 1}	10	1.00	0.69
4	{1, 1, 1, 1, 1, 1, 1, 1, 1, 6}	15	0.50	0.67
5	{1, 1, 1, 1, 1, 1, 1, 1, 6, 6}	20	0.50	0.64
6	{2, 2, 2, 2, 2, 2, 2, 2, 2, 12}	30	0.50	0.50
7	{1, 1, 1, 1, 1, 1, 2, 3, 3, 6}	20	0.62	0.73
8	{1, 1, 1, 3, 3, 3, 3, 4, 5, 6}	30	0.78	0.78

[1] D.M. Chiu and R. Jain, "Analysis of the increase and decrease algorithms for congestion avoidance in computer networks," Computer Networks and ISDN Systems, vol.17, pp.1–14, 1989.

提案指標の特徴

- ネットワーク全体のスループットが考慮されている
- 利用率と公平性，両方を含めて評価することができる

まとめ

- TCPのACKパケット損失の検出時にも輻輳制御を適用することで公平性を向上
- 上り通信フロー間，上り通信フローと下り通信フローとの間　両方で改善がみられた

- 公平性と利用率を考慮した評価指標の提案