インターネットフローの公平性

大阪大学
長谷川 剛
hasegawa@ics.es.osaka-u.ac.jp

目次

- 背景
- インターネットフロー間の不公平
- 公平性向上のための手段
  - プロトコルの改善
  - ルータバッファでの制御
  - エンドホストでの資源管理
- まとめ

背景

- これまでのインターネット
  - ベストエフォートネットワーク
  - QoS保証は行われない
    - 帯域、遅延等
    - TCPを使えば、「確実に相手に届く」だけは何とかなる
  - 繋がることが重要だった
    - 当然、ユーザー間、フロー間の公平性は期待できない

背景

- バックボーン、アクセス回線速度の飛躍的な向上
  - 「繋がる」以上のサービスへの要求
  - QoS保証
- 「公平性」が重要な要素になりつつある
  - 同じ料金なのに速度が違う
  - 2倍のアクセス速度を2倍の料金で使っているのにスループットは同じ

公平性の定義

- *場所によって異なる*
  - 同じ輻輳ルータを通るフローは同じスループット
  - アクセス回線速度に応じたスループット
- *フローとは？*
  - 個別のTCPコネクション
  - ユーザ単位

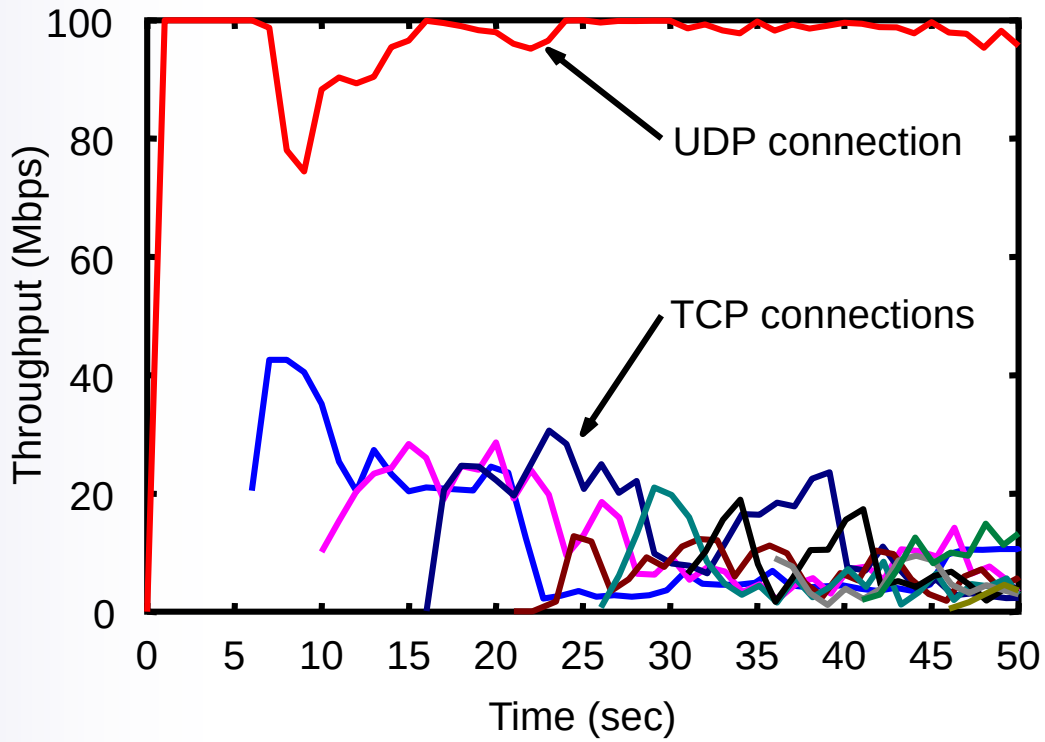
インターネットフロー間の不公平

- プロトコルの違い
  - TCPとUDP
  - TCPのバージョン
- TCPコネクションの環境の違い
  - 遅延（距離）
  - 帯域
  - …

TCPフローとUDPフロー

- UDPは輻輳制御を行わない
  - ネットワークの輻輳に関係なく一定のレートでパケットを転送
- TCPフローとUDPフローが混在すると
  - 輻輳時にTCPフローは転送レートを下げる
  - UDPフローが帯域を占有
- TCPが輻輳制御を行わないように改良（改悪？）することも容易

TCPフローとUDPフロー



- TCPフローが増えても、UDPフローのスループットは下がらずに帯域を占有

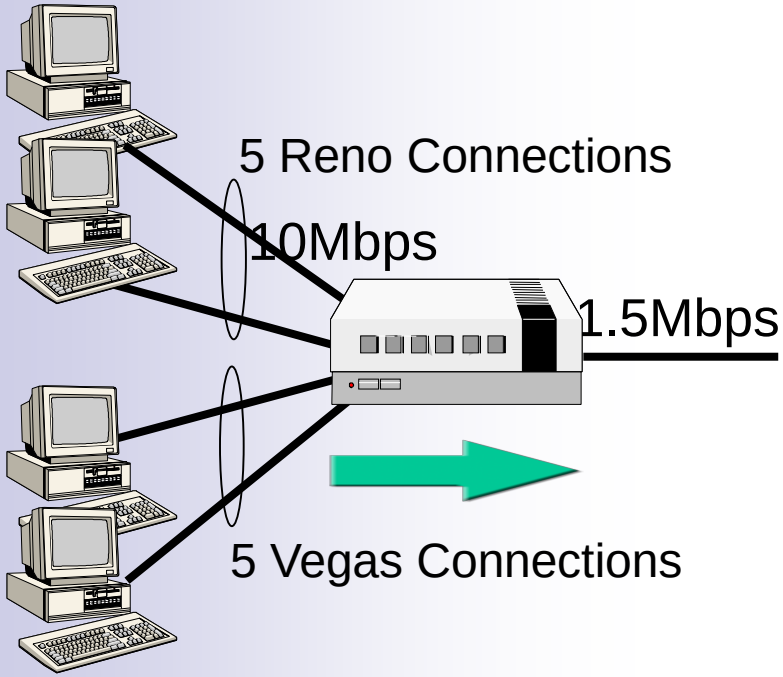
TCPのバージョン

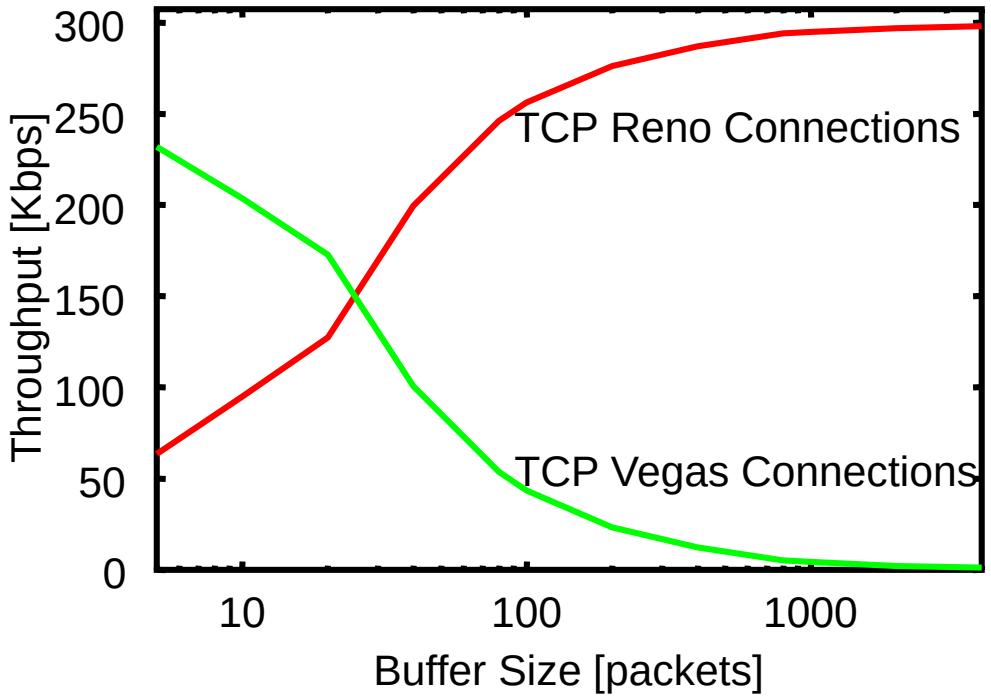
- TCP Tahoe, TCP Reno
  - 現在のほとんどのOSの実装ベース
  - パケットロスのみで輻輳を検知
- OSによって実装はバラバラ
  - タイマ処理
  - ウインドウサイズの初期値
  - スループットに大きく影響

TCPのバージョン

- *TCP Vegas*
  - RTT (Round Trip Time) の大小で輻輳を検知
  - Tahoe/Renoに比べて高いスループットが得られる
- *RenoとVegasの差異*
  - Renoは積極的に転送レート (ウィンドウサイズ) を上げ、パケットロスを前提とした制御を行う
  - VegasはRTTの増加をパケットロスの前兆ととらえ、パケットロスが発生しないように制御を行う

TCPのバージョン





- 混在すると、性能がいいはずのVegasが積極的なRenoに負ける

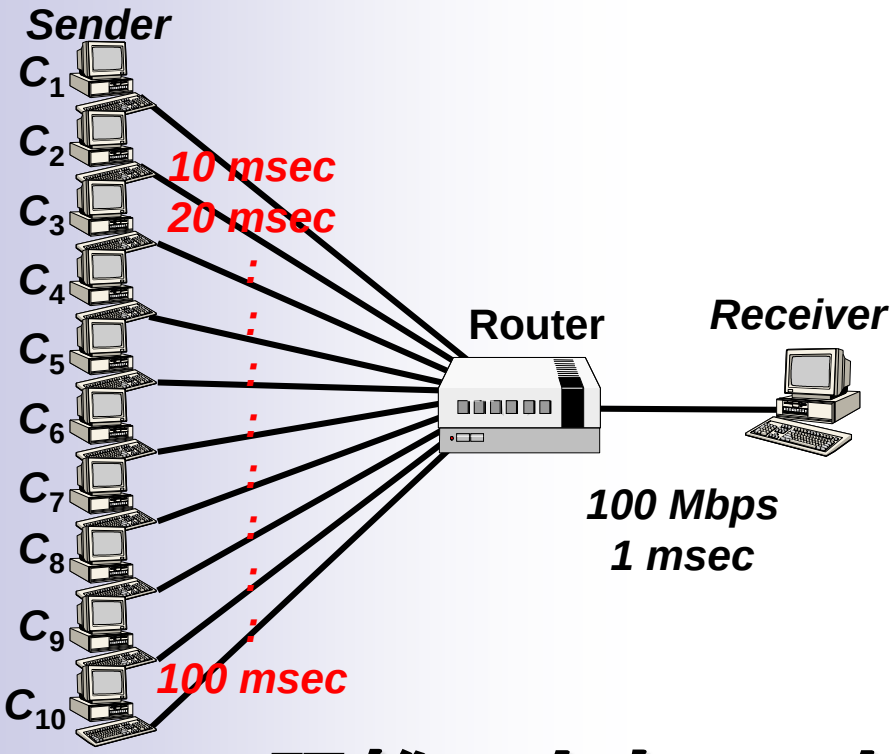
距離（伝播遅延時間）の違い

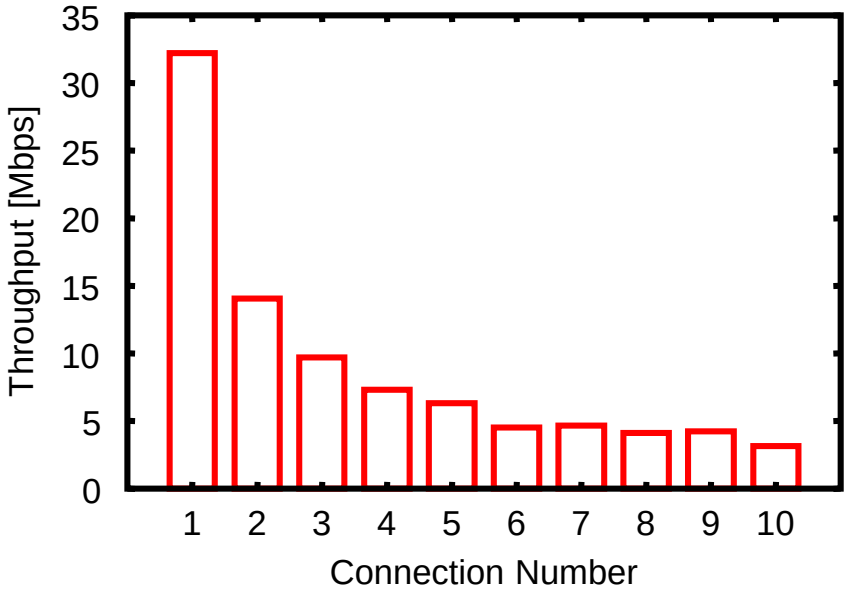
- TCPのスループットは送受信間の距離に大きく影響される

$$\rho = \max\left(\frac{W_{\max}}{RTT}, \frac{1}{RTT\sqrt{\frac{2bp}{3}} + T_0 \min\left(1, 3\sqrt{\frac{3bp}{8}}\right) p\left(1+32p^2\right)}\right)$$

J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP throughput: a simple model and its empirical validation," in *Proceedings of ACM SIGCOMM'98*, pp. 303–314, August 1998.

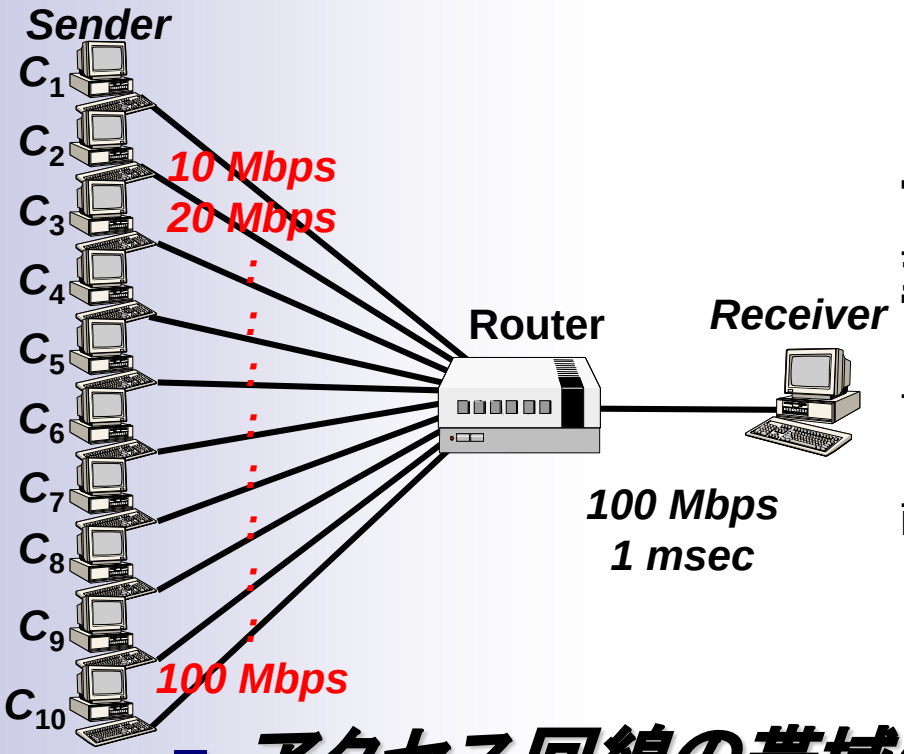
遅延時間の違い

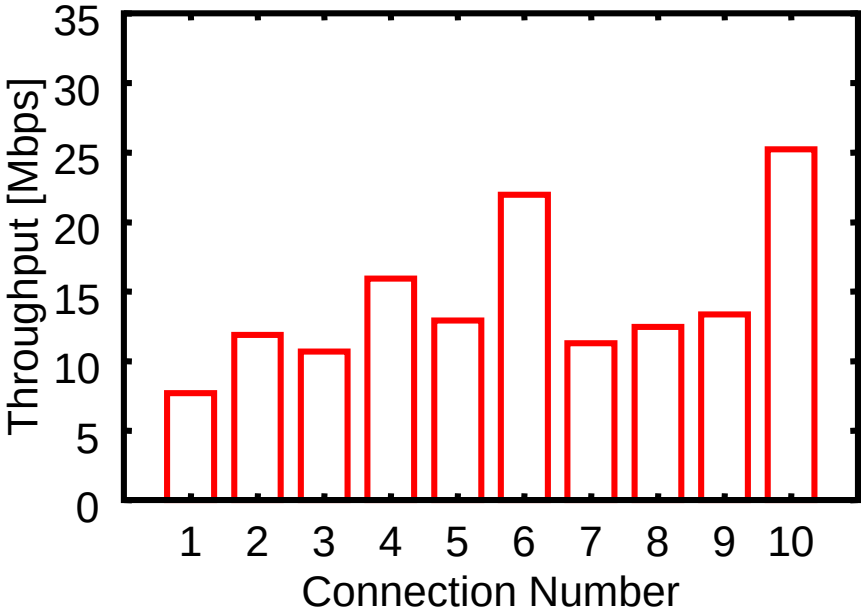




- 距離の大きいコネクションは非常に不利

帯域の違い





- アクセス回線の帯域の違い
  - スループットは等しくならない
  - アクセス回線の帯域に比例もしない

その他

- ホップ数
  - 通過する輻輳ルータの数が多いほどスループットは下がる
- 転送するファイルサイズの差
  - 小さいファイルを転送するコネクションのスループットは小さくなる
  - Slow Startが原因
- 送受信端末のバッファサイズ
  - コネクションの状態に関係なく固定値が割り当てられる
- ...

公平性改善の方法

- 送信側端末での制御
  - UDPフローのTCP-friendly制御
  - TCPプロトコルの改善
- ルータバッファでの制御
  - AQM (Active Queue Management)
- エンドホストでの資源管理

TCP-friendly制御

- TCP-friendlyとは
  - UDPフローは、同一パスを通るTCPフローと平均のスループットが同じであるべき
  - アプリケーションが制御を行う
  - 2つの制御方法
    - TCPスループット推測の式を利用する
    - TCPと同様のAIMD (Additive-Increase Multiplicative Decrease) 制御を行う
- TCPとの公平性を向上

TCPプロトコルの改善

- *TCPの本質的な欠点*
  - フロー制御とエラー制御（再送制御）の区別が非常にあいまい
  - パケット単位の制御と、バイト単位の制御が混在している
- *改善方式*
  - TCP Vegasなどさまざまな方式が提案
  - TCPに変わる全く新しいプロトコルを考えるのは非現実的
  - 段階的な実装、新旧TCP間の公平性を考慮する必要がある

ルータバッファでの制御

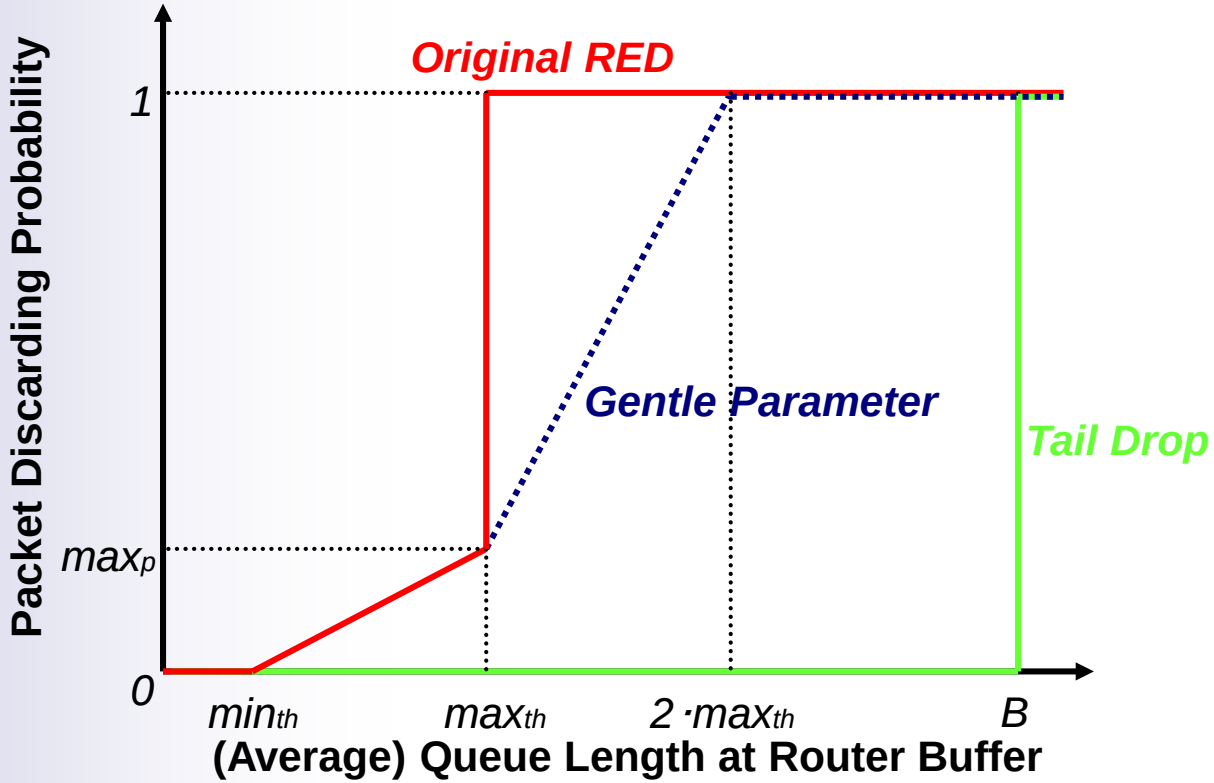
- *Tail-Drop Router*
  - 単純なFIFOバッファで、実装が容易
  - バッファが一杯になると到着パケットを廃棄
  - 問題点
    - バースト的なパケットロス
    - TCP のタイムアウトを引き起こしやすい
  - 公平性は全く期待できない

AQM (Active Queue Management)

- TCPはネットワークをブラックボックスと考えている
  - 異環境の下での高い公平性は期待できない
- ルータバッファで積極的な制御を行う
  - パケットの確率的な廃棄
  - 複数キューを用いたフロー管理

RED (Random Early Detection)

- バッファが一杯になる前に、パケットを確率的に廃棄
- 廃棄率はキュー長によって変動



RED (Random Early Detection)

- TCPのスループット、公平性向上
  - バースト的なパケット廃棄が減少
  - TCPのタイムアウトが減少
- 問題点
  - パラメータ設定の難しさ
  - TCPの本質的な不公平性は解消されない
- 非常に多くの改善方式が提案されている

REDの改善方式

- FRED (Flow Random Early Detection)
  - ルータ内パケットをフロー毎にカウント
- SRED (Stabilized RED)
  - 確率的な動作をコネクション数予測に利用
- CSFQ (Core-Stateless Fair Queuing）
  - エッジルータとコアルータで協力
- DRED, ARED, BLUE, GREEN,...
- 制御の複雑さと公平性はトレードオフ
  - どれを使うかは適用箇所の要求条件による

ECN (Explicit Congestion Notification)

- ルータバッファでのパケット廃棄を避ける
  - 輻輳時にルータでパケットを廃棄するのではなく、マーキングを行う
  - 送信側TCPはマークされていればパケットロスと同様に扱う
- TCPの不公平性が解消されるわけではない
  - 送信側の動きは定義されていない
  - すべてのルータ、エンドホストが対応する必要がある

複数キューを用いた制御

- 古くから提案されている
  - RR (Round Robin), FQ (Fair Queuing)
- 高い公平性
- 問題点も多い
  - 制御の重さ
  - フロー衝突
- DRR (Deficit Round Robin)
  - O(1)でできる
  - パケットサイズの違いも考慮
- やはり、制御の度合いと公平性はトレードオフ

エンドホストでの資源管理

- TCPの送受信バッファ
  - 固定値が割り当てられている
  - バッファサイズで最大レートは制限
  - 小さすぎるとネットワークに余裕があるときに帯域を使い切れない
  - 大きすぎるとネットワークが混雑しているときにバッファが無駄
- 特に繁忙なWWWサーバ等で無駄が生じやすい

エンドホストでの資源管理

- 割り当て資源量を動的に制御する
  - ATBT
    - バッファサイズをウインドウサイズに応じて増減
  - SSBT
    - バッファサイズを推定TCPスループットに応じて増減
  - 公平に、かつ必要なところへ必要なだけ資源を配分
- 資源の浪費の回避、サーバ能力の向上

まとめ

- TCPは既にインターネットに広く普及しているため、全く新しいトランスポート層プロトコルを提案することは非現実的
- 既存のTCP との親和性、及び提案方式を段階的に適用した場合の影響を考慮すべき

まとめ

- フロー間の公平性を向上させるためには、エンドホストにおける輻輳制御だけではなく、ネットワークルータにおける何らかの制御が必要
- ルータにおいて公平性を向上させる機構を持たせる際には、公平性の定義とその程度、収容するフロー数等によって、適切な複雑さを持つアルゴリズムを選択して適用すべき