



# 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

大阪大学 大学院基礎工学研究科  
情報数理系専攻 博士前期課程  
牧 一之進



# 発表内容

---

- 研究の背景
- 研究の目的
- 階層化パケットスケジューリング方式の提案
- 評価モデル
- シミュレーションによる評価
- まとめと今後の課題

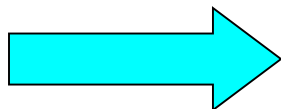


# 研究の背景

- インターネットのインフラ化
- ユーザのアクセス帯域の増加



- 特定のユーザのみが大きな帯域を占有




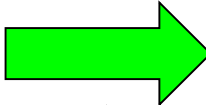
公平なインターネットの構築  
各ユーザフローを公平にサービス



# 従来の研究

- CSFQ (Core Stateless Fair Queueing)
  - エッジルータでレートを測定してパケットヘッダに書きこむ
  - コアルータでそのレートをもとに動的に廃棄確率を決定する

 ヘッダの拡張が必要であり、すべてのエッジルータを更新する必要がある。
- DRR (Deficit Round Robin)
  - フロー毎にキューを設けてスケジューリング

 コアルータでもすべてのフローに対してキューを設ける必要があるので実装が困難



# 研究の目的

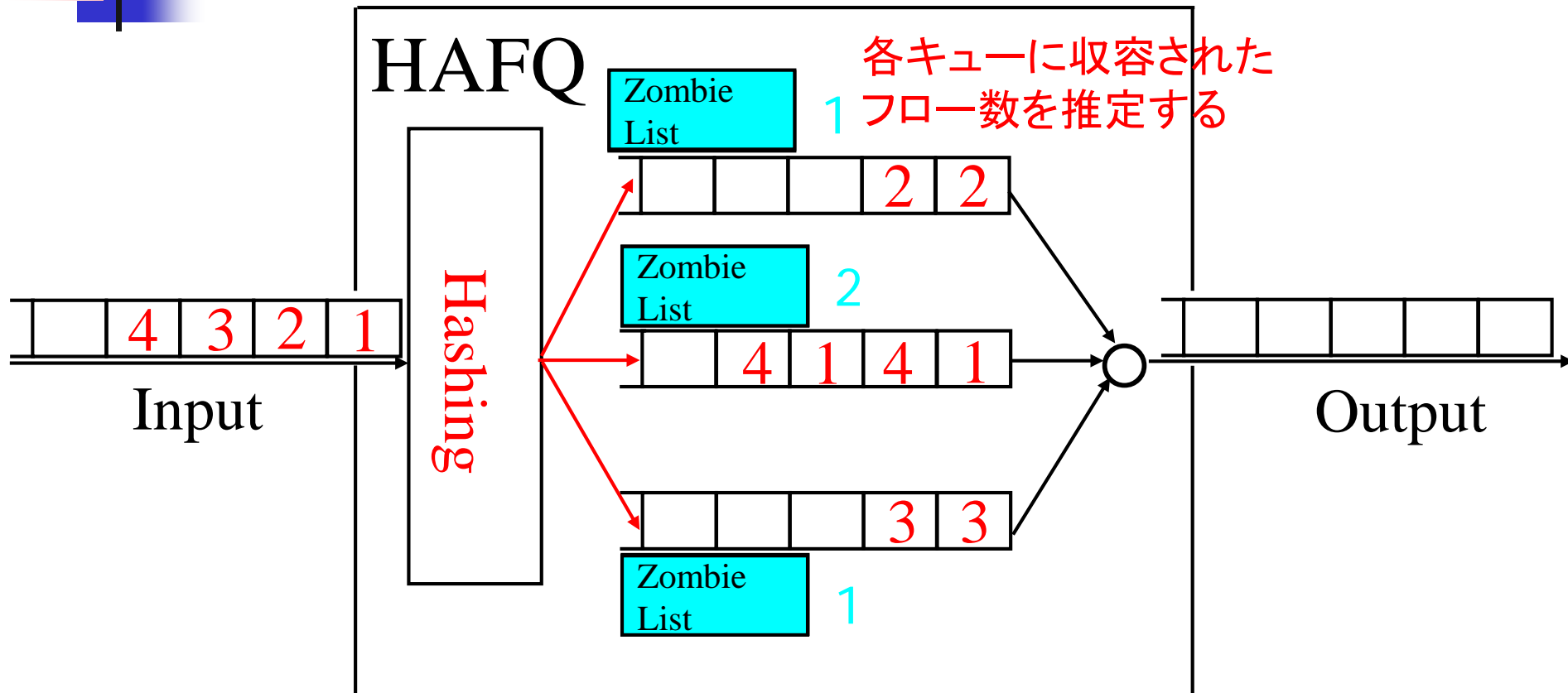
- ヘッダの拡張がなくフロー毎の情報をもたないパケットスケジューリング方式の提案
  - 基本的にDRRのようなper-flowのスケジューリング
  - 扱うべきフロー数にしたがってフローを集約



エッジからコアまで適用できてスケーラブル



# 提案方式(Hierarchically Aggregated Fair Queuing)の概要(1/3)




各フローのパケットを振り分ける



# 提案方式(Hierarchically Aggregated Fair Queuing)の概要(2/3)

ゾンビリスト : {Flow Id, counter}を  
1つの組とする  
過去に到着したフロー  
に関する履歴

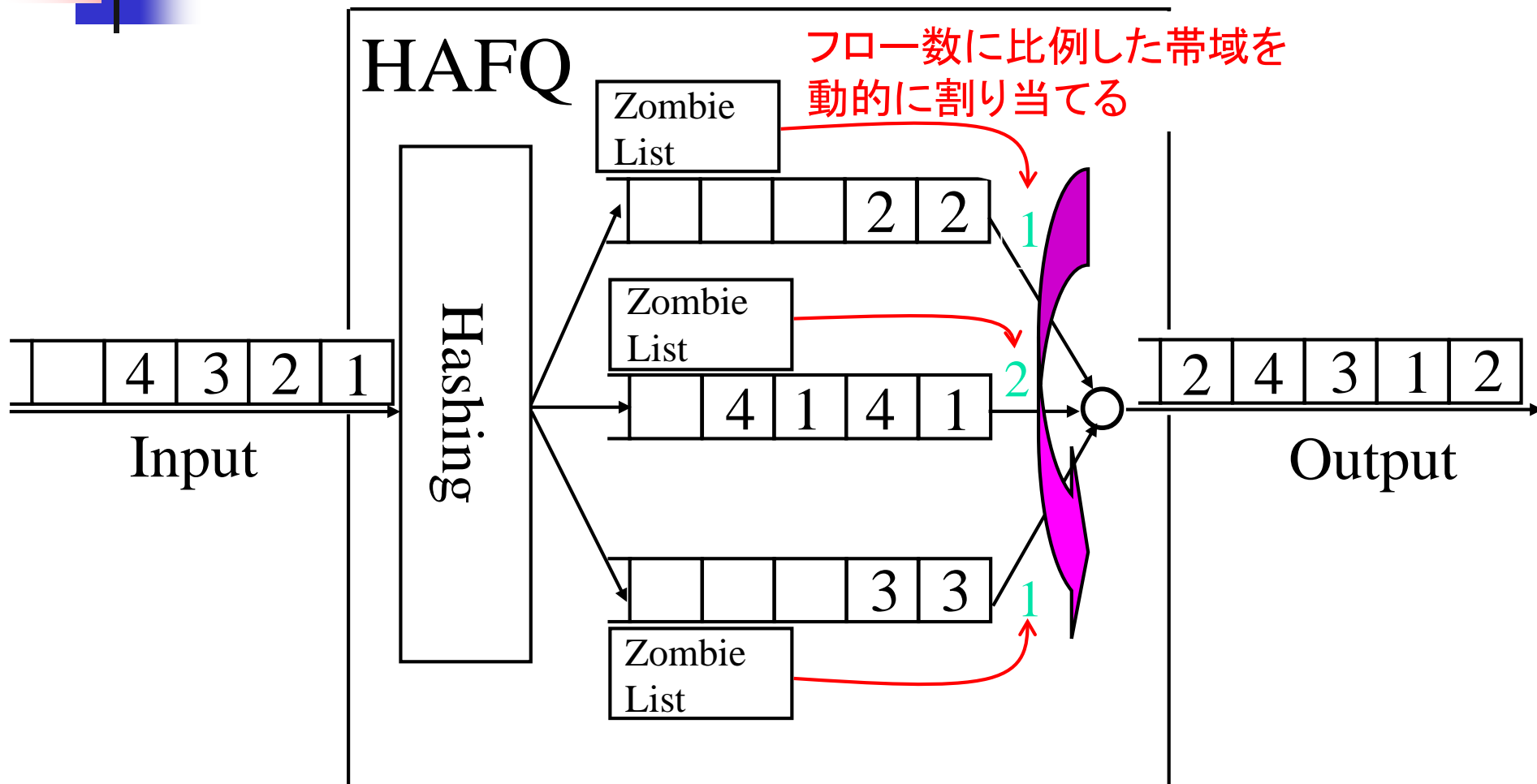
Flow Id	counter
1	3
2	7
5	1
7	4

 すべてのフローの情報は必要ない

- そのキューに收容されたフロー数を推定する
- 同一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する



# 提案方式(Hierarchically Aggregated Fair Queuing)の概要(3/3)

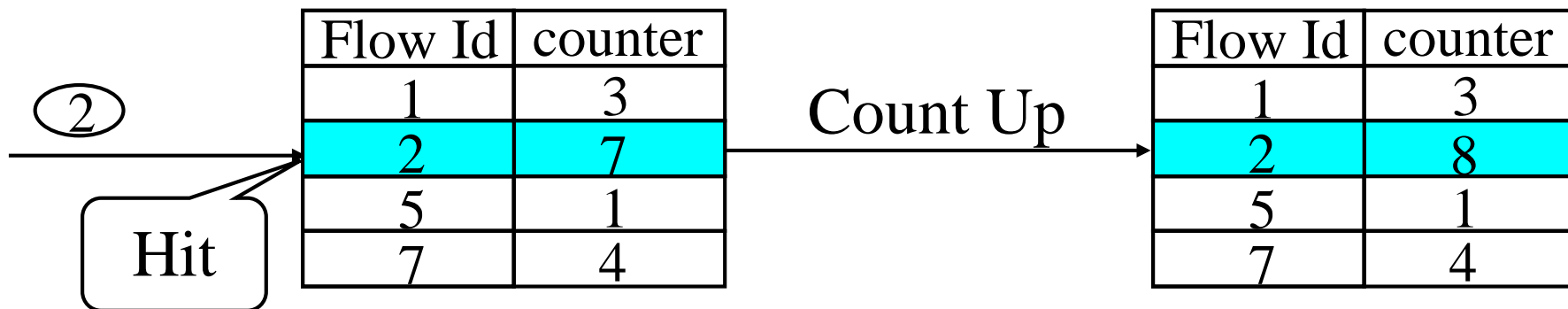






## ゾンビリスト(リスト内にIDがあるとき)

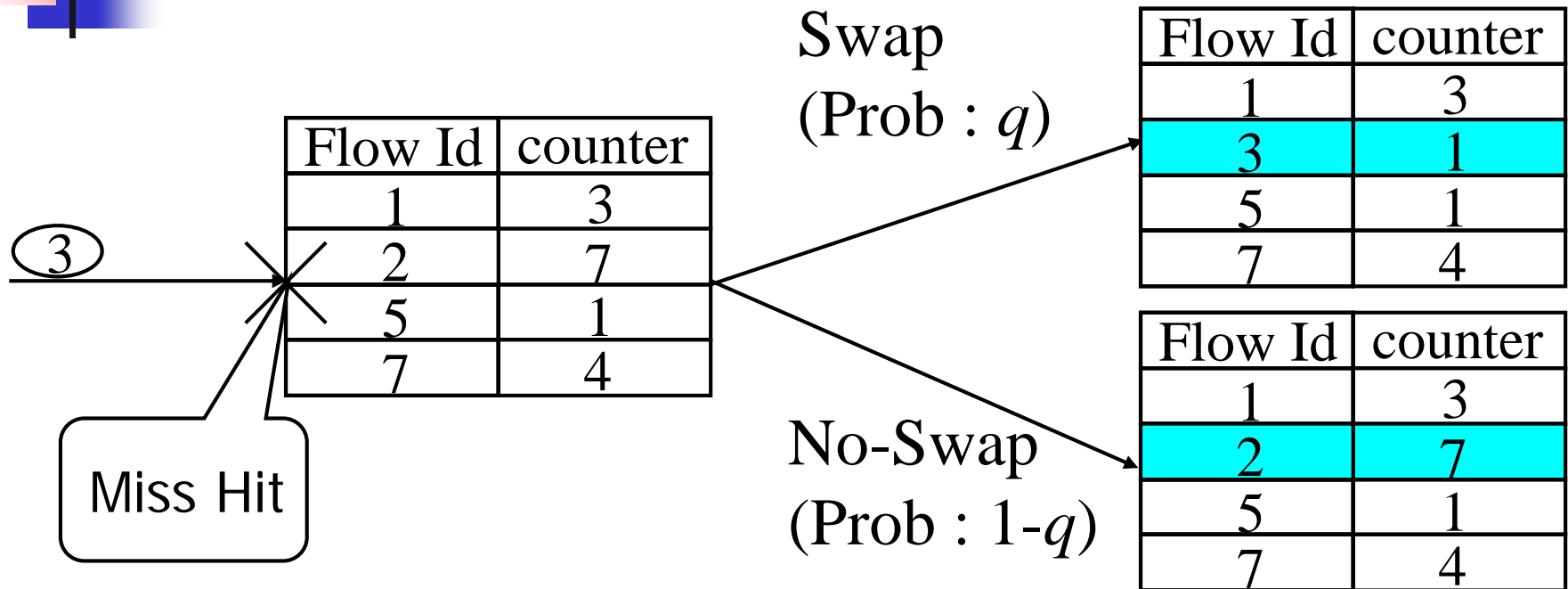
- パケットがルータに到着したときのゾンビリストの動作
    - ゾンビリストをすべて検索する
- ➡ 少ないエントリ数で、できる限り多くのフローを管理する



ゾンビリスト内に到着して来たフローのIDがあれば、そのゾンビのカウンタ値を1増やす



## ゾンビリスト(リスト内にIDが存在しないとき)



ゾンビリスト内に到着して来たフローのIDがなければ、 $q$  の確率で置き換え、カウンタ値を1に初期化する。  
 $1-q$  の確率で何もしない。



# SREDのフロー数推定アルゴリズム

SREDのフロー数推定方式

各フローのレートがほぼ一定であると仮定

比較の際、同じ  
である確率：  
 $p$  (ヒット率)

②

Flow Id
2
5
7

2のある確率:  $1/N$   
 $N$ : フロー数

$$N = 1/p$$

レートにかたよりのある場合、うまくいかない



## 提案方式のフロー数推定アルゴリズム(1/2)

$$\lambda_{avg} = \sum_{i=1}^N \lambda_i / N \quad (1) \quad \lambda_i : \text{フロー}i \text{のレート}$$
$$N = \sum_{i=1}^N \lambda_i / \lambda_{avg} \quad \lambda_{avg} : \text{全フローの平均レート}$$
$$N : \text{フロー数}$$

フロー数 = 全到着レート / 全フローの平均レート



レートにかたよりのある場合にもフロー数を正しく推定できる

$R_i$  を全到着レートに占めるフロー $i$ のレートの割合として、 $R_{avg}$  から推定フロー数を求める

$$\text{推定フロー数} = \frac{1}{R_{avg}}$$



## 提案方式のフロー数推定アルゴリズム(2/2)

$R_i$  の計算は、ゾンビの内容が置き換えられるときに行う  
→ このときのカウンタ値が、そのフローのレートに比例

### 問題点

■ レートの高いフローが存在する場合、他のフローよりも多く平均到着割合に組み入れられる

→ 平均をとるさいに、重みをつける

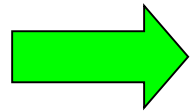
■ フロー数がエントリ数より少ない場合、定常状態でカウンタ値が発散してしまう

→ エントリ数を動的に減らし、常にフロー数がエントリ数より大きくなるようにする



# カウンタによる廃棄

- 各キュー間の公平性は保たれる



ところが同一キュー内の公平性は保たれない

- ゾンビ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



カウンタ値が大きいフローの packets を優先的に廃棄

Flow Id	counter
1	3
2	2
5	14
7	2



# 評価モデル(シングルリンクモデル)

フロー数: 64本

帯域 : 155 [Mbps]

伝播遅延時間 : 1 [ms](bottleneck link)

0.1 [ms](access link)

Sender Hosts

Receiver Hosts

S1

R1

Bottleneck Link

Router

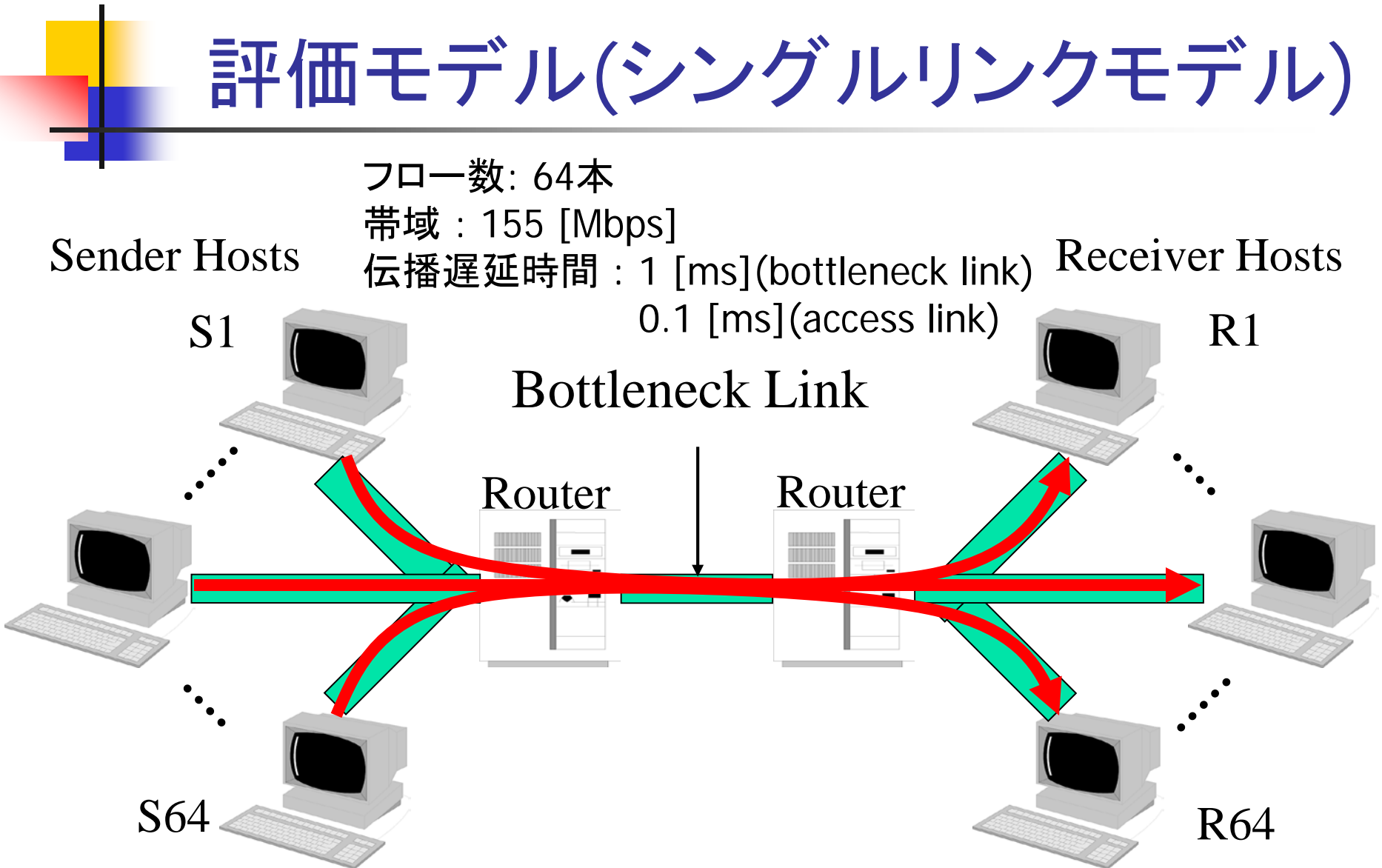
Router

S64

R64

2001/6/22

ネットワークシステム研究会





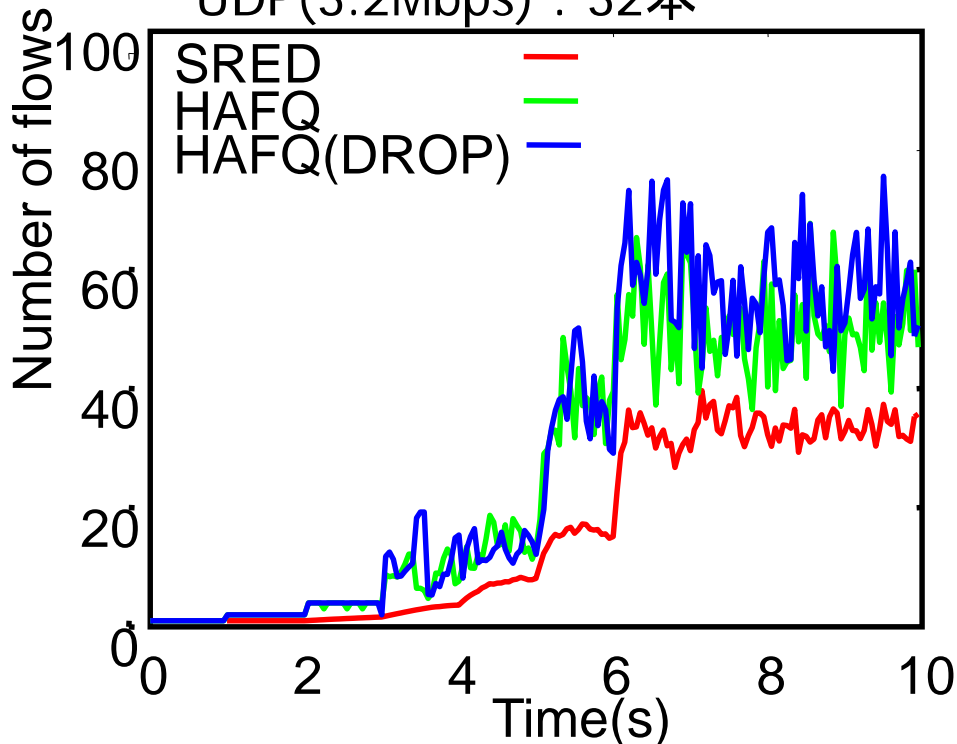
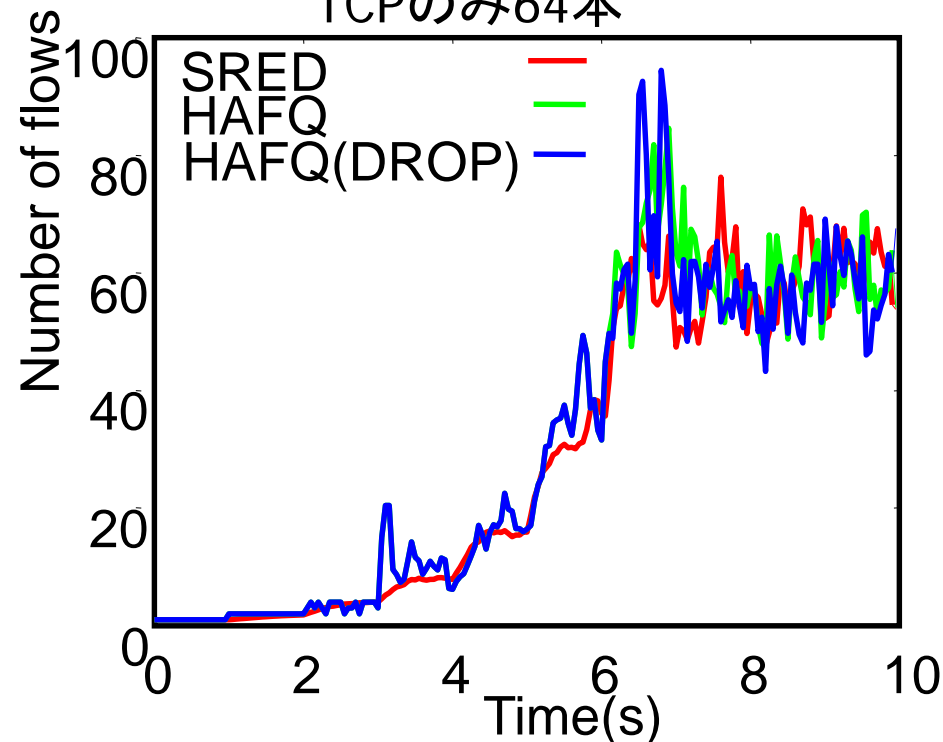
# 推定フロー数の評価

- 1秒毎にフロー数を2倍にしていく、キュー数を1とし、ゾンビ数を4とする

TCP : 32本

UDP(3.2Mbps) : 32本

TCPのみ64本



**SREDに比べてHAFQは推定フロー数が実際の数に近い**

2001/6/22

ネットワークシステム研究会



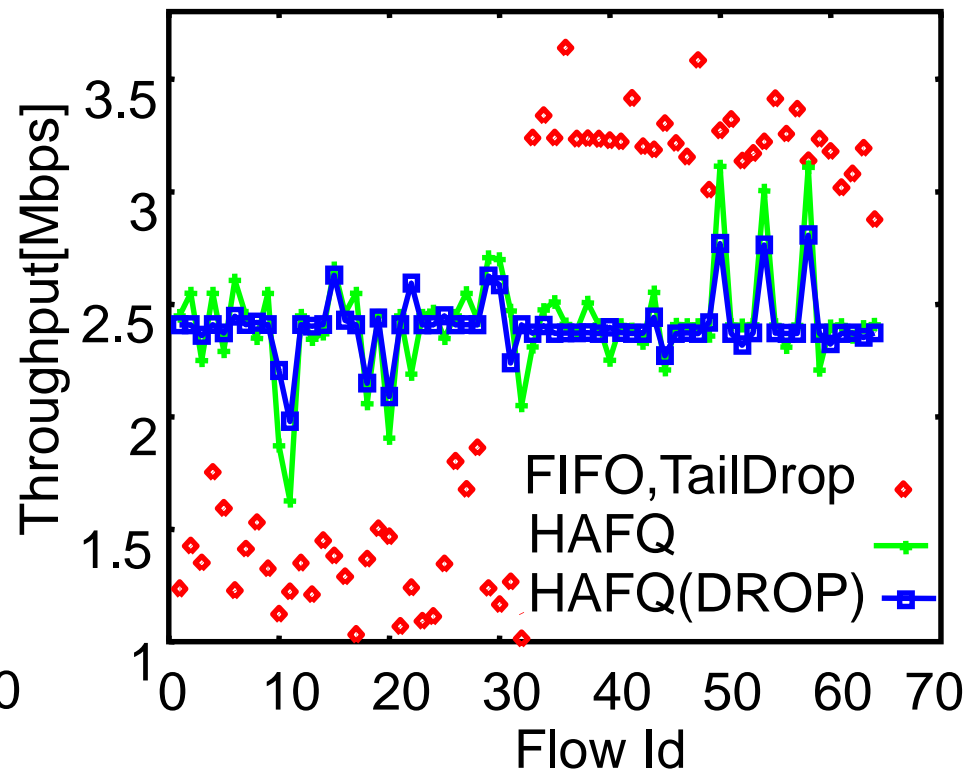
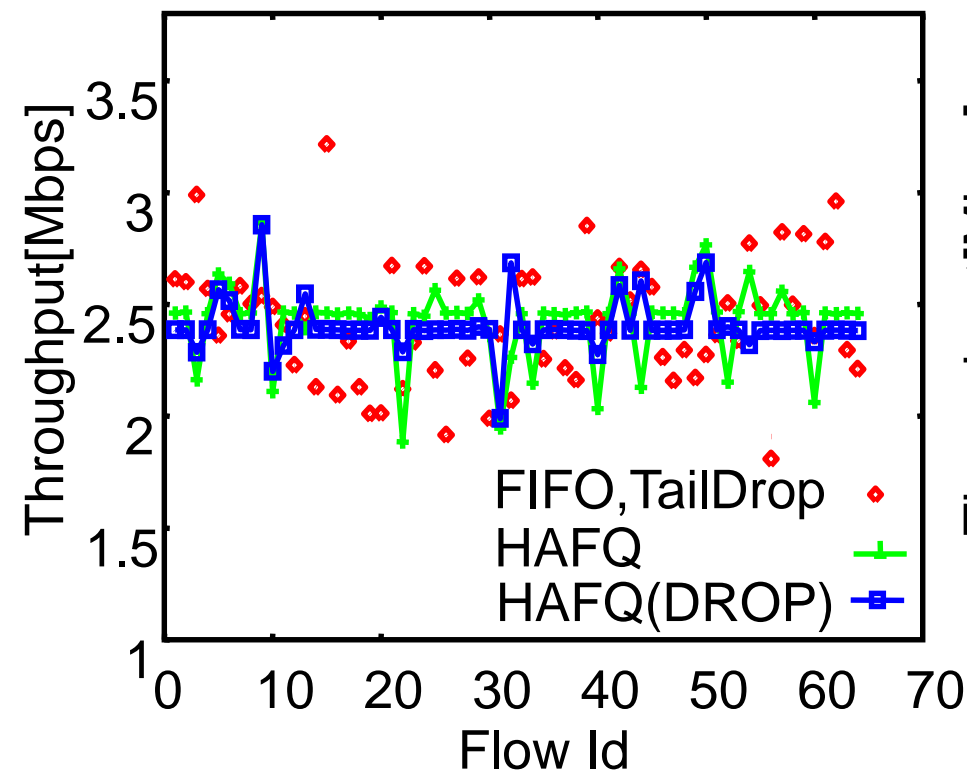


# 各フローのスループットの評価

● 各フローは同時に送信を開始する  
TCPのみ64本

TCP : 32本

UDP(3.2Mbps) : 32本



レートの高いフローが存在する場合、HAFQ(DROP)では高い公平性を実現



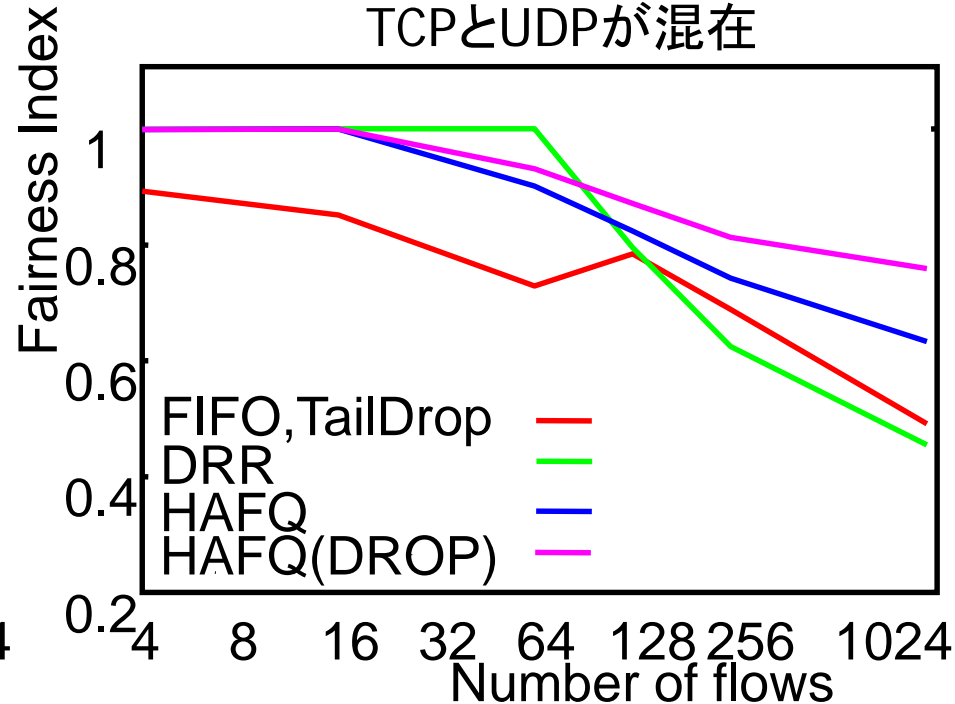
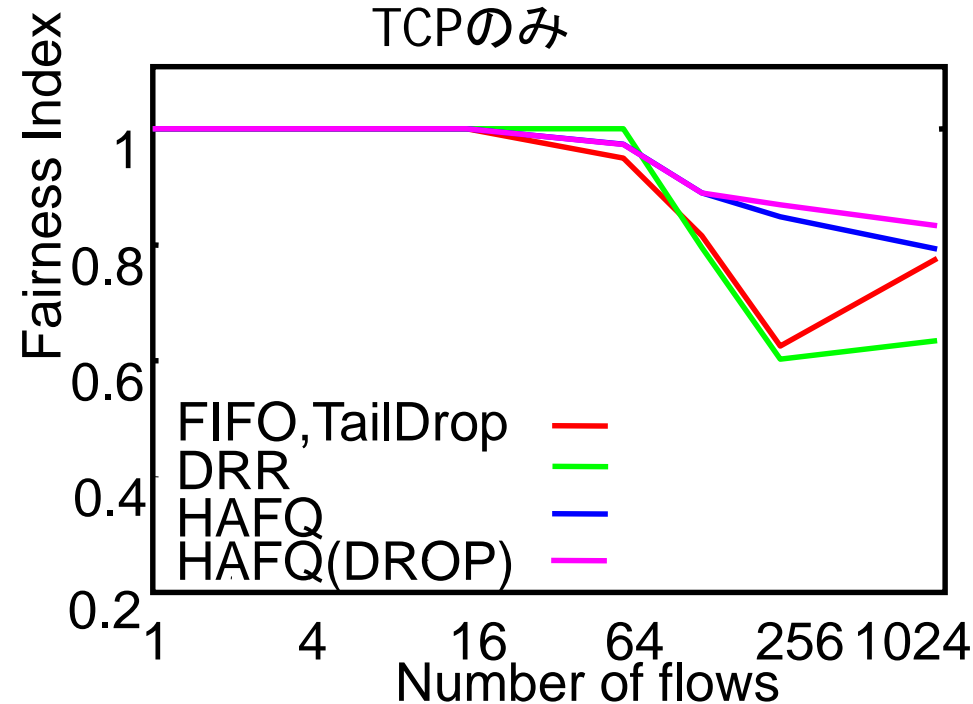
# フロー数を変化させた場合の評価

DRR: キュー数64とし、フロー数が64本を越えるとき、ランダムにキューイング

HAFQ: CRC16でハッシング  
キュー数64,ゾンビ数4

TCPのみ

TCPとUDPが混在



HAFQ: DRRと同等の性能(フロー数が少ないとき)

従来手法に比べて高い公平性を実現(フロー数が多いとき)



# まとめと今後の課題

## ■ まとめ

- エッジルータやコアルータの能力に応じてスケーラブルに実装可能なパケットスケジューリング方式の提案
- フロー毎の優れた公平性を実現

## ■ 今後の課題

- ルータを多段に配置した場合の影響
- 既存のルータと共存できるのか？