



# 公平なネットワーク利用を実現する スケーラブルな パケットスケジューリング方式

*Hierarchically Aggregated Fair Queuing (HAFQ)  
for Per-flow Fair Service in High-speed Networks*

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2年  
牧 一之進



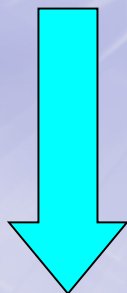
# 発表内容

- **研究の背景**
- **研究の目的**
- **提案方式 (HAFQ)の説明**
- **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

- 従来はアクセス回線がボトルネック
  - バックボーンへの流入トラヒックの制限によりユーザ間の公平性はあまり問題ではない
- ユーザのアクセス帯域の増加  
ADSL , FTTHなど

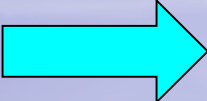


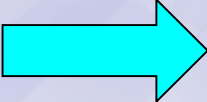
- 特定のユーザのみが大きな帯域を占有  
→ 公平なインターネットの構築  
各ユーザフローを公平にサービス



# 従来の研究

- *CSFQ (Core Stateless Fair Queueing)*
  - Edge: 各フローのレートをパケットヘッダに書き込む
  - Core: 各フローのレートを元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)*
  - フローごとにキューを設けてスケジューリングを行う

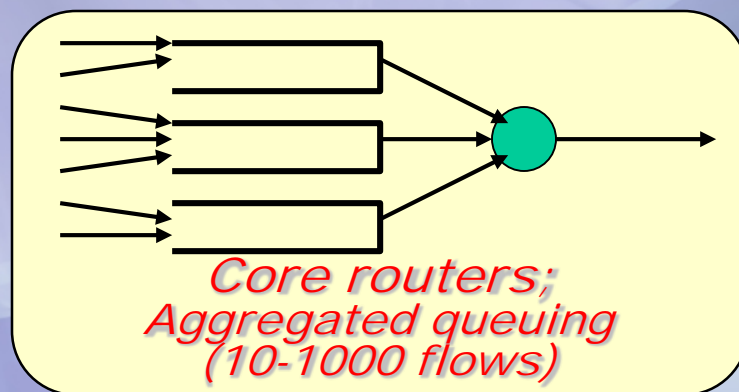
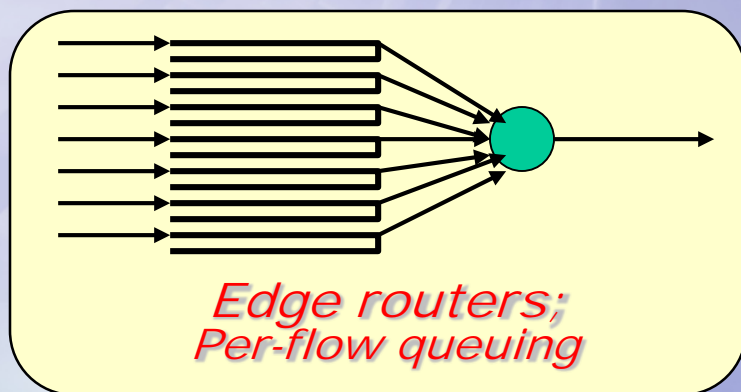
 **コアルータでもすべてのフローに対してキューを設ける必要がある**

# 研究の目的

- ヘッダの拡張がなくフローごとの情報を持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

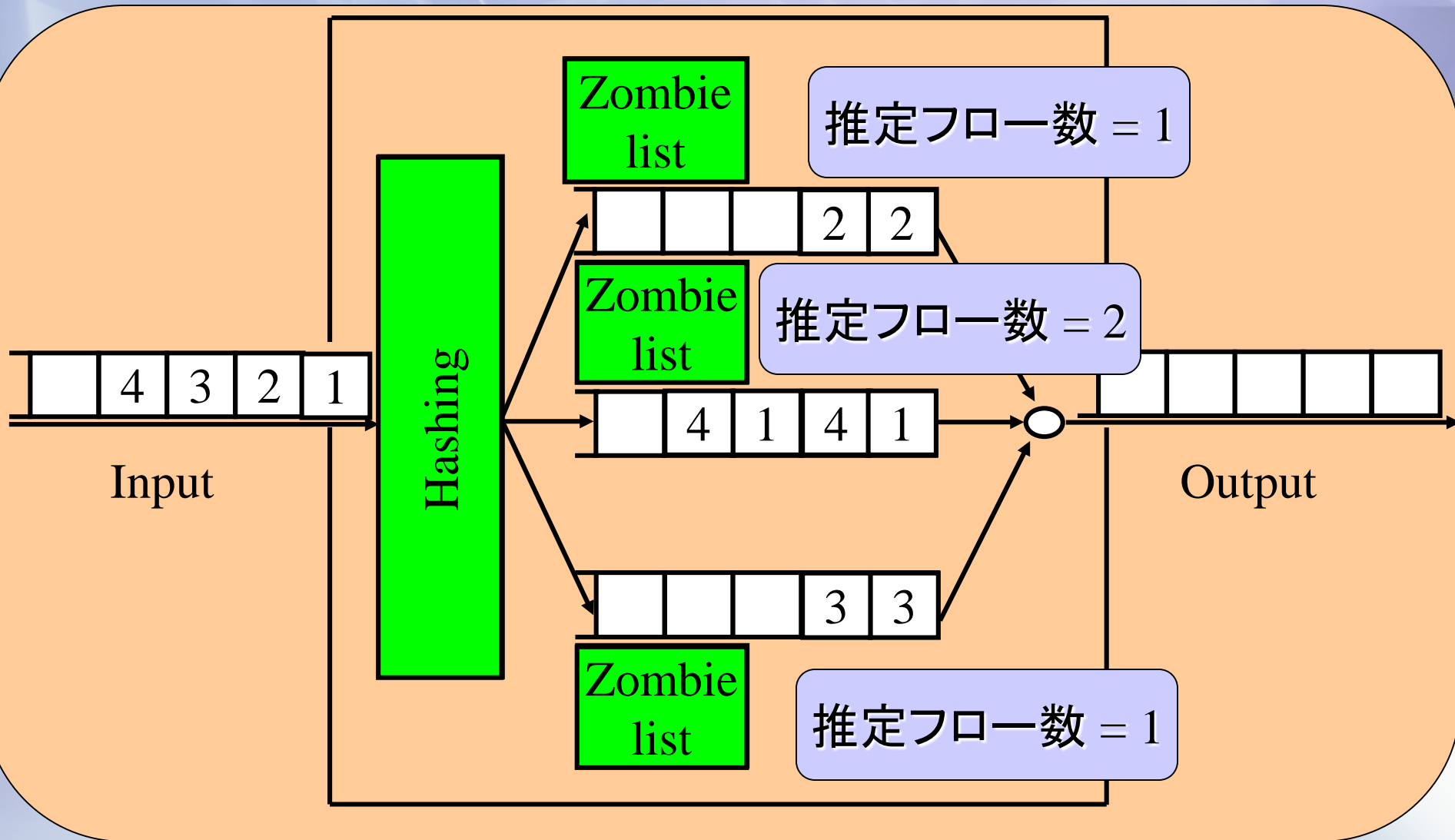


エッジからコアまで適用できてスケラブル





# 提案方式の概要





# ゾンビリストの概要

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	2
7	4

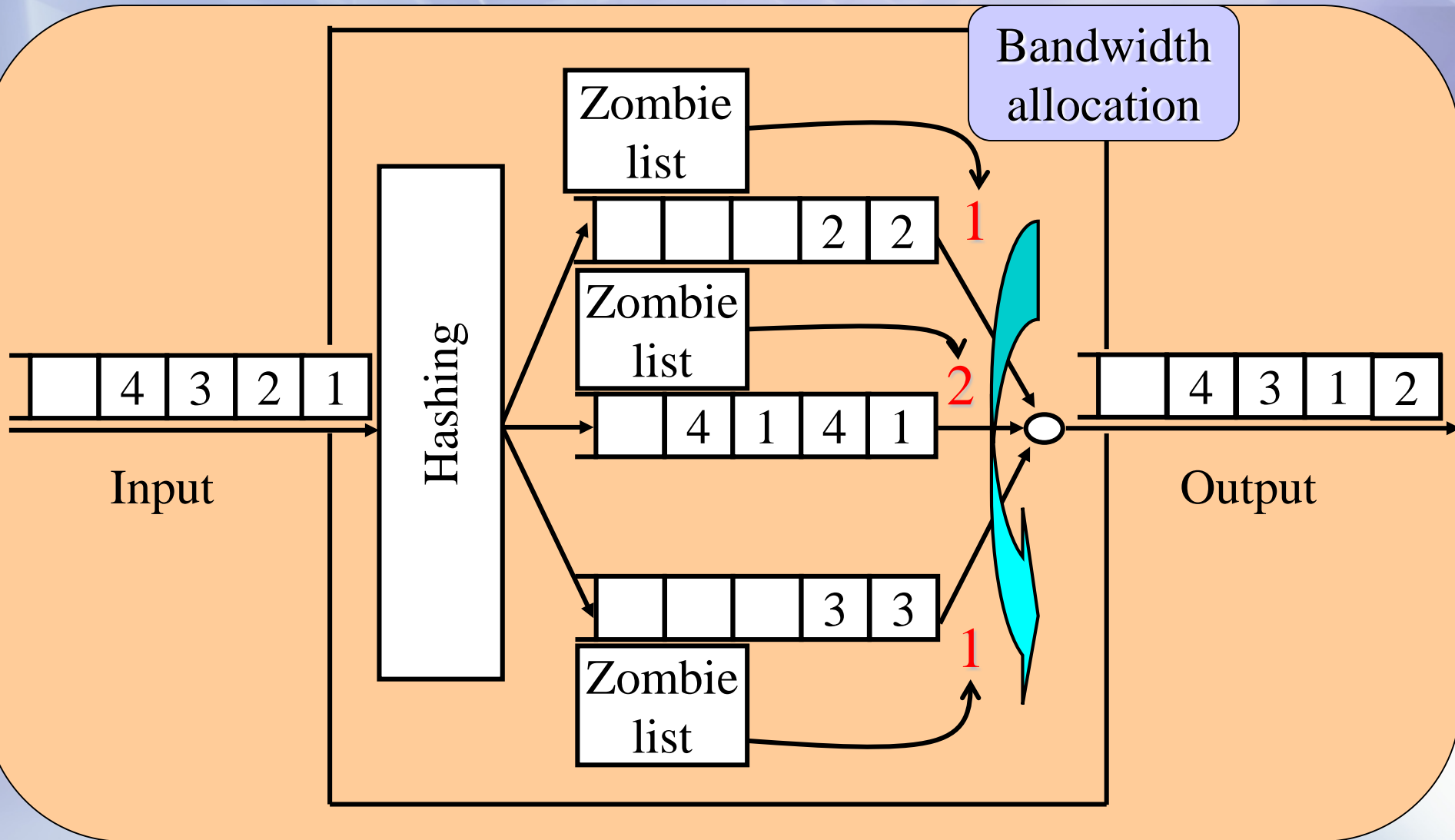


すべてのフローの情報はいらない

- 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



# 提案方式の概要

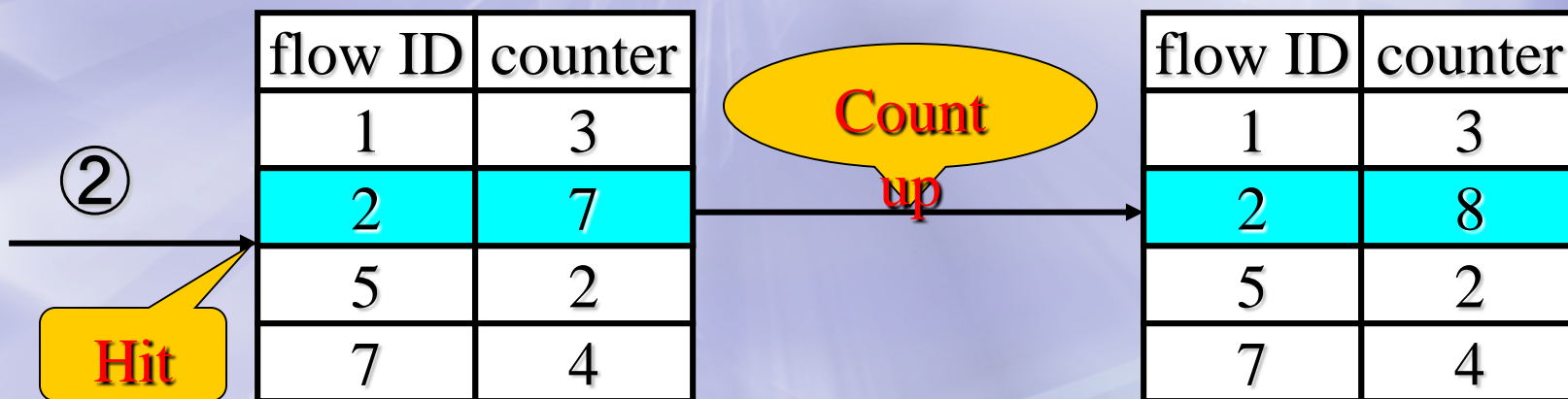






# ゾンビリスト (リスト内にIDがあるとき)

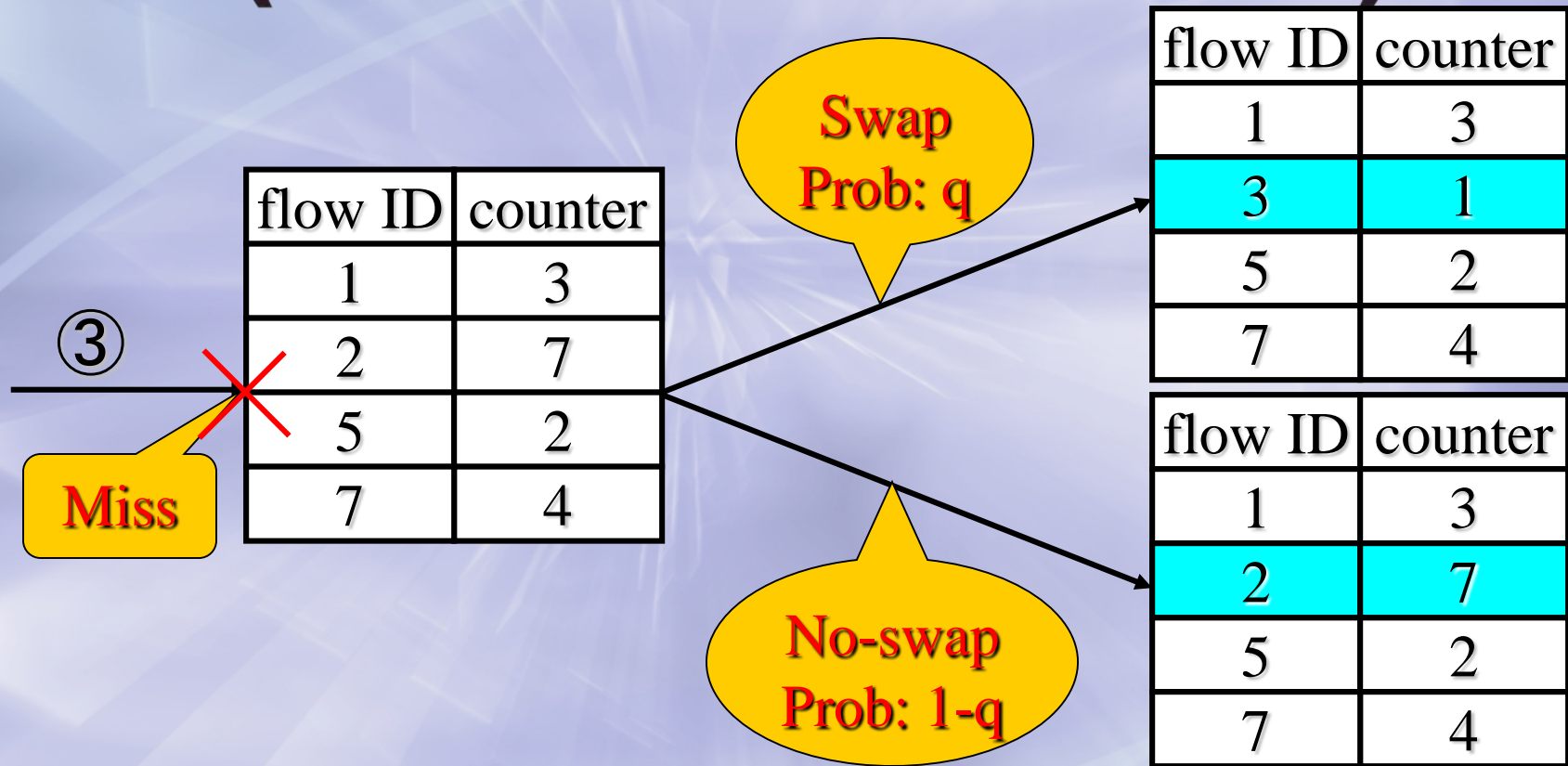
- パケットがルータに到着したときの動作
  - ゾンビリストをすべて検索する
  - 少ないエントリ (ゾンビ) 数で、できる限り多くのフローを管理する



ゾンビリスト内に到着したきたフローの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がないとき)



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



# フロー数推定方式

$$\text{全フローの平均到着率} = \frac{\text{全フローの到着率の合計}}{\text{フロー数}}$$



$$\text{フロー数} = \frac{\text{全フローの到着率の合計}}{\text{全フローの平均到着率}}$$

※各フローの到着率に偏りがある場合も成立



## ●フロー数推定アルゴリズム

- Step1: 到着パケットが属するフローの到着率を計算
- Step2: 全フローの到着率の平均を計算
- Step3: 推定フロー数を計算



# フロー数推定方式のアルゴリズム

## ■Step1: 到着パケットが属するフローの到着率を計算

- ゾンビリストのカウンタ最大値より到着率を統計的に推定
  - カウンタ最大値はフローの到着率に比例
  - ゾンビがSwapされたときのみ計算
  - 推定到着率 = (1-ヒット率)置き換え確率 / ゾンビ数(カウンタ最大値-1)

## ■Step2: 全フローの到着率の平均を計算

- 推定到着率の移動時間平均を計算
  - 平均到着率 =  $a$ 平均到着率 + (1- $a$ )計算した推定到着率
- レートの高いフローは高い割合で平均到着率に組み入れられる
  - $a$ にバイアス $\Rightarrow a = \beta \times (\text{カウンタ最大値} / \text{推定レート})$

## ■Step3: 推定フロー数を計算

- 推定フロー数 =  $1 / \text{平均到着率}$



# 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
  - 例: 同一キュー内にレートの高いUDPとTCPが存在する場合
- 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している



カウンタ値が大きいフローの packets を優先的に廃棄

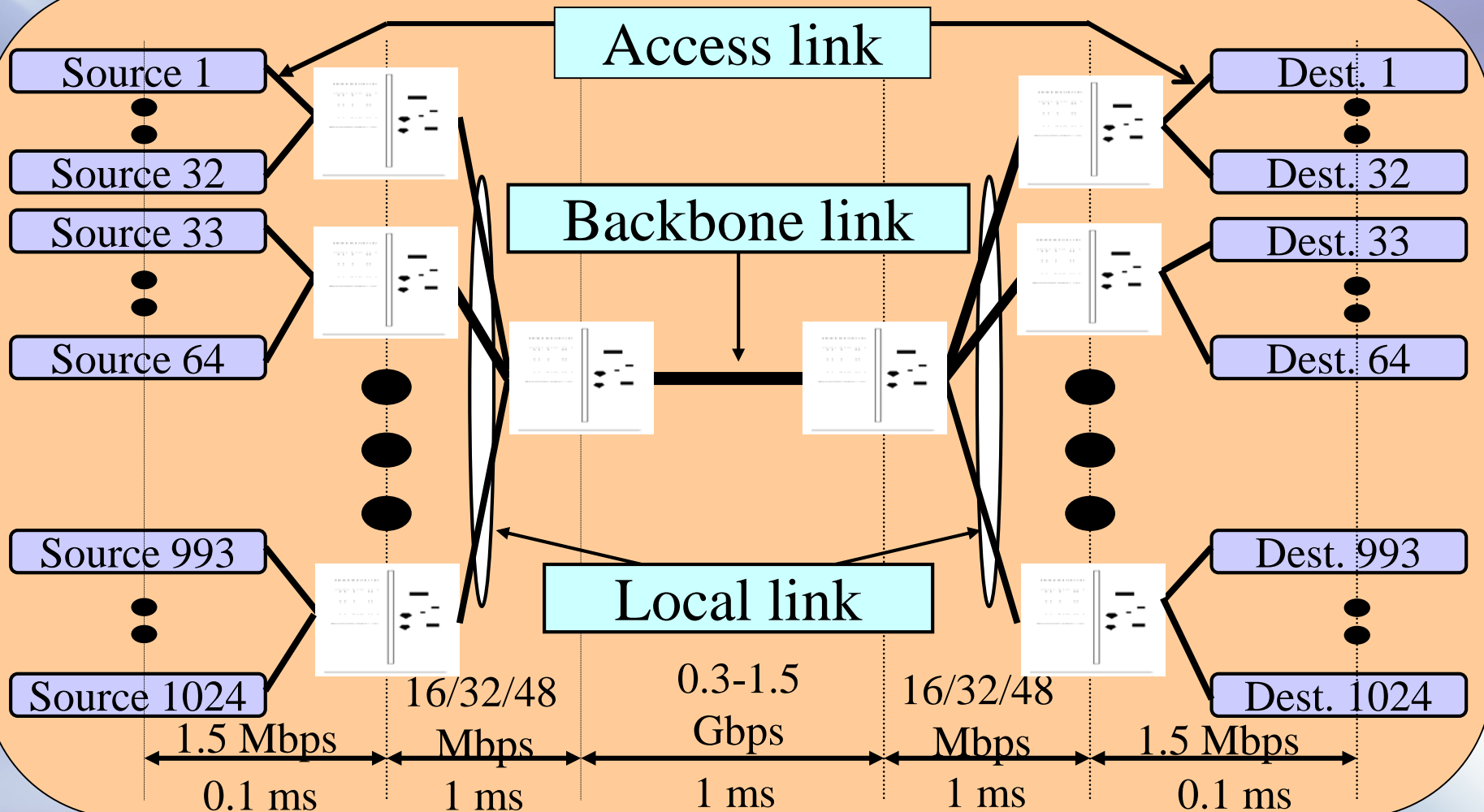
flow ID	counter
1	2
2	10
5	2
7	3

ある期間内に  
ルータに到着した  
パケット数



# 評価モデル

Edge router : TCP (26), UDP (6) ; UDP : 3.2Mbps(CBR)

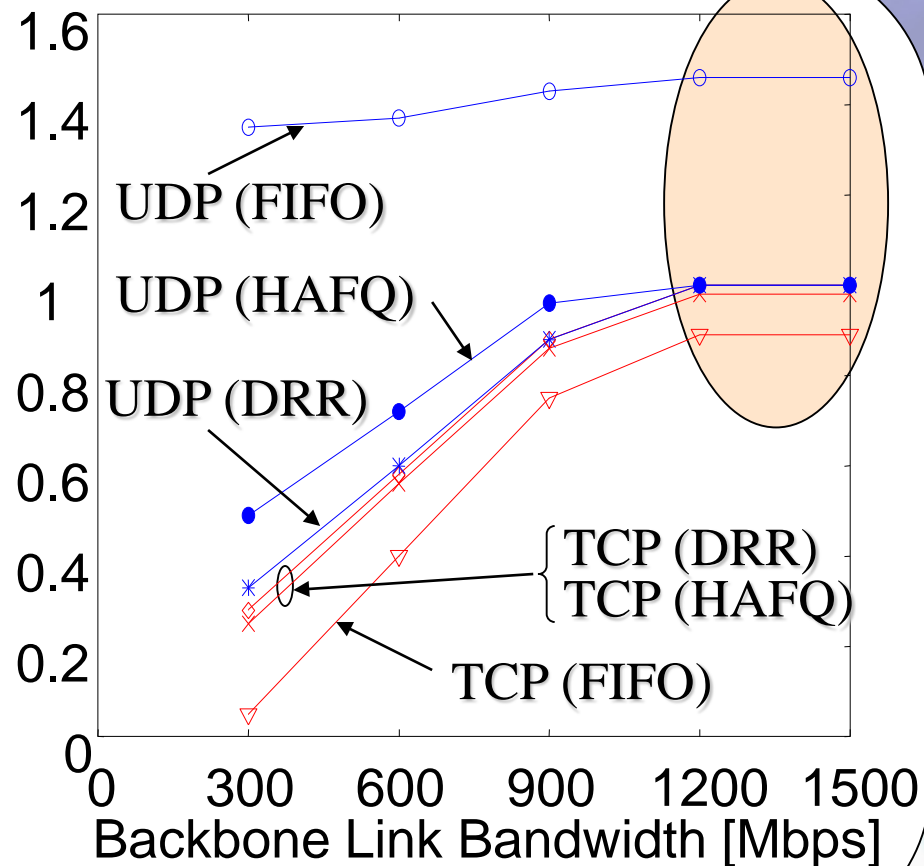
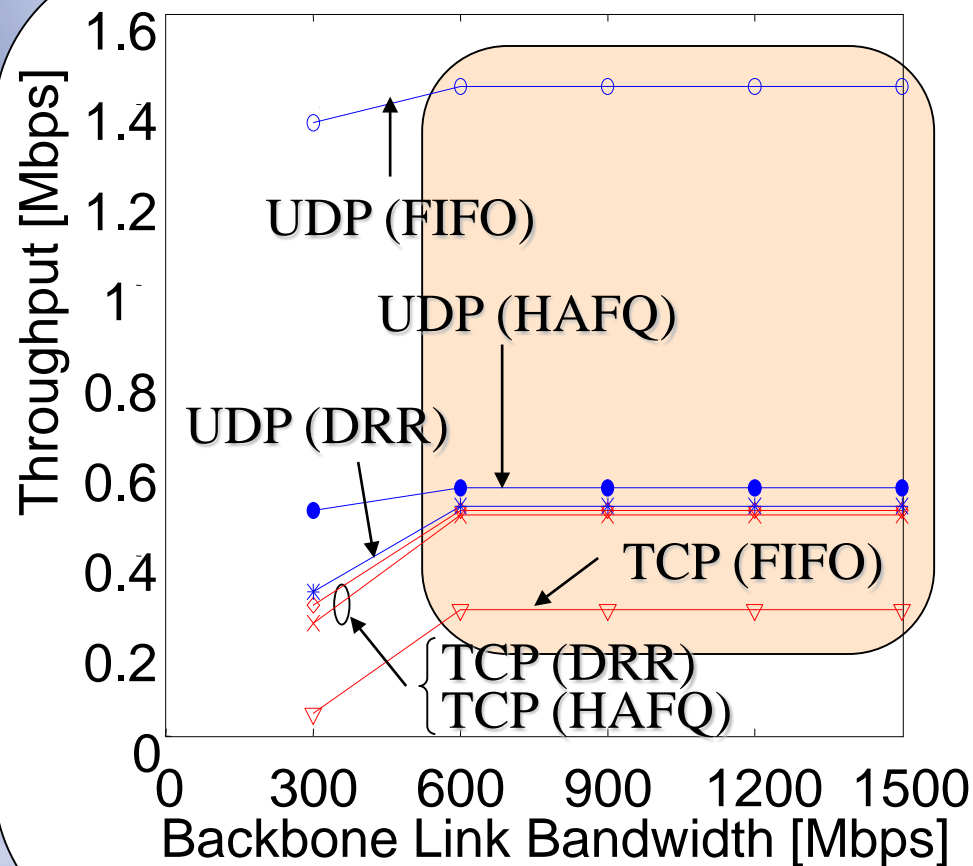


# エッジルータがボトルネックになっている場合



Local link 16Mbps

Local link 32Mbps



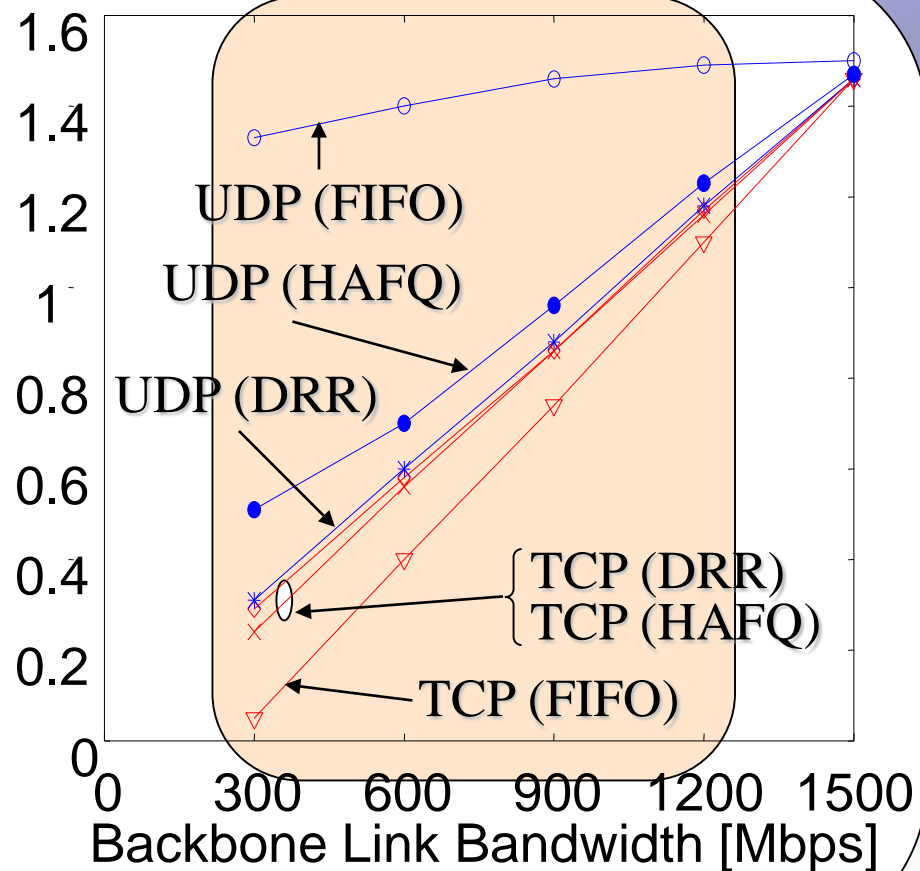
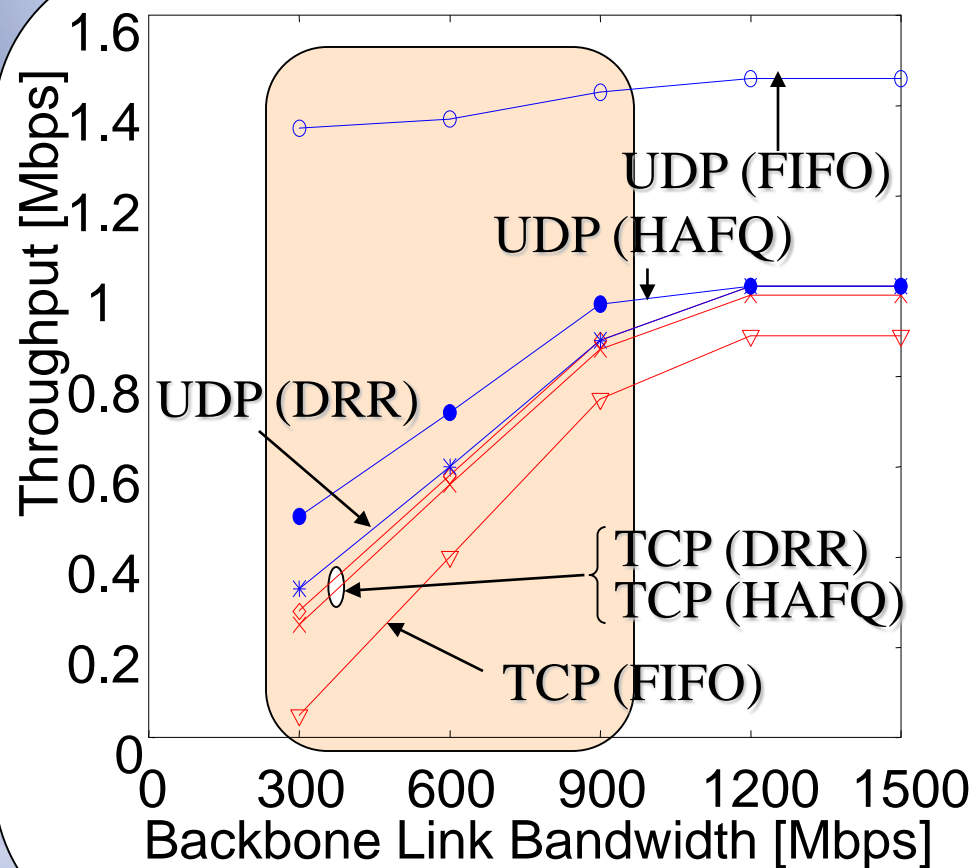
**DRR, HAFQではTCPとUDPの間の不公平性が解消**

# コアルータがボトルネックに なっている場合



Local link 32Mbps

Local link 48Mbps



DRRはフロー間の不公平性を解消  
HAFQもDRRに近い公平性

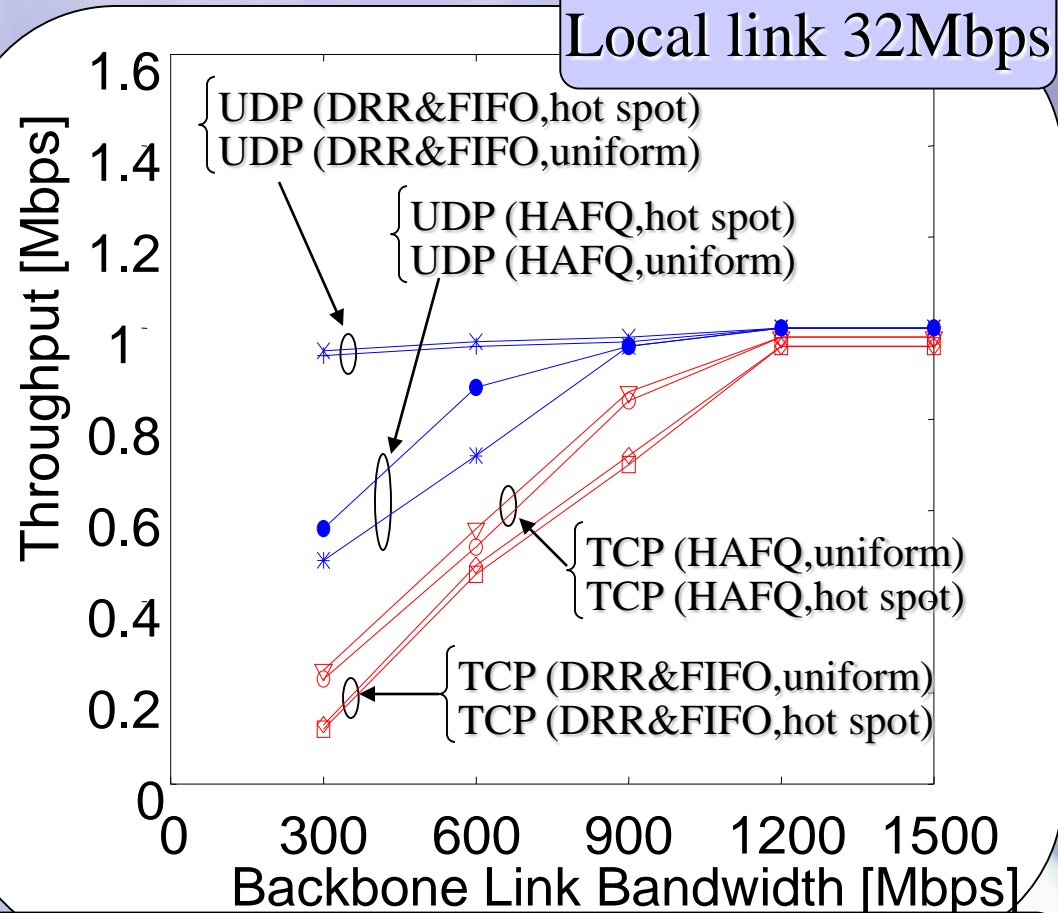




# コアルータでフローごとの キューイングが困難な場合

**Hot spot** : 特定のエッジルータに  
UDPを收容

**Uniform** : 各エッジルータに  
バランスよくUDPを收容



エッジルータのみでのper-flowキューイングでは公平性を維持できない  
HAFQはフロー間の公平性が向上



# 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**

- 既存のルータと共存できるのか？