

The background features abstract green geometric shapes. On the left, a solid green triangle points downwards. On the right, a complex arrangement of overlapping translucent green triangles and polygons creates a layered, dynamic effect. The central text is positioned between these two main graphic elements.

# 情報ネットワーク

伊藤 嘉浩

# 講義内容

- ▶ 1) 序論
- ▶ 2) 3) ネットワークアーキテクチャ
- ▶ 4) 5) 伝送路と物理層
- ▶ 6) 誤り制御方式
- ▶ 7) MACプロトコル
- ▶ 8) 中間試験
- ▶ 9) データリンク層プロトコル
- ▶ 10) 11) データ交換とネットワーク層
- ▶ 12) 13) 14) 15) TCP/IP
- ▶ 16) 期末試験

# 11) データ交換と ネットワーク層(2)



# データ交換と ネットワーク層

経路制御（ルーティング）の分類

# ルーティング

## ルーティングアルゴリズムの分類

### 1. 経路指定に関して

#### ▶ ホップバイホップルーティング

- ▶ 各ルータで経路制御方に基づき経路を決定

#### ▶ ソースルーティング

- ▶ 送信者が途中の経路（一部・全部）を指定

※赤字が本講義の対象

### 2. 制御方式に関して

#### ▶ 非適応型

- ▶ 現在のトラフィックやトポロジーを考慮せず静的に制御
- ▶ Flooding（洪水）, 最短パス(Shortest Path)

#### ▶ 適応型

- ▶ トラフィックやトポロジーの変化に応じて経路を制御
  - ▶ 中央制御（centralized）
  - ▶ 孤立適応（isolated） 例）ホットポテトルーティング
  - ▶ 分散制御（distributed） 例）距離ベクトル型, リンク状態型

# 代表的な分散制御方式

## ▶ 距離ベクトルルーティング

- ▶ 各ルータは**隣接ルータとのみ**周期的に**距離ベクトル**（ルーティング情報）を交換
- ▶ 各ルータは、得られた情報を基にして他のルータの各々に対して一組の記載事項（好ましい出線、そのルータまでの**“距離”**（ホップ数や遅延の推定値など））を持つルーティング表を作成
- ▶ 例）**RIP**, IGRP

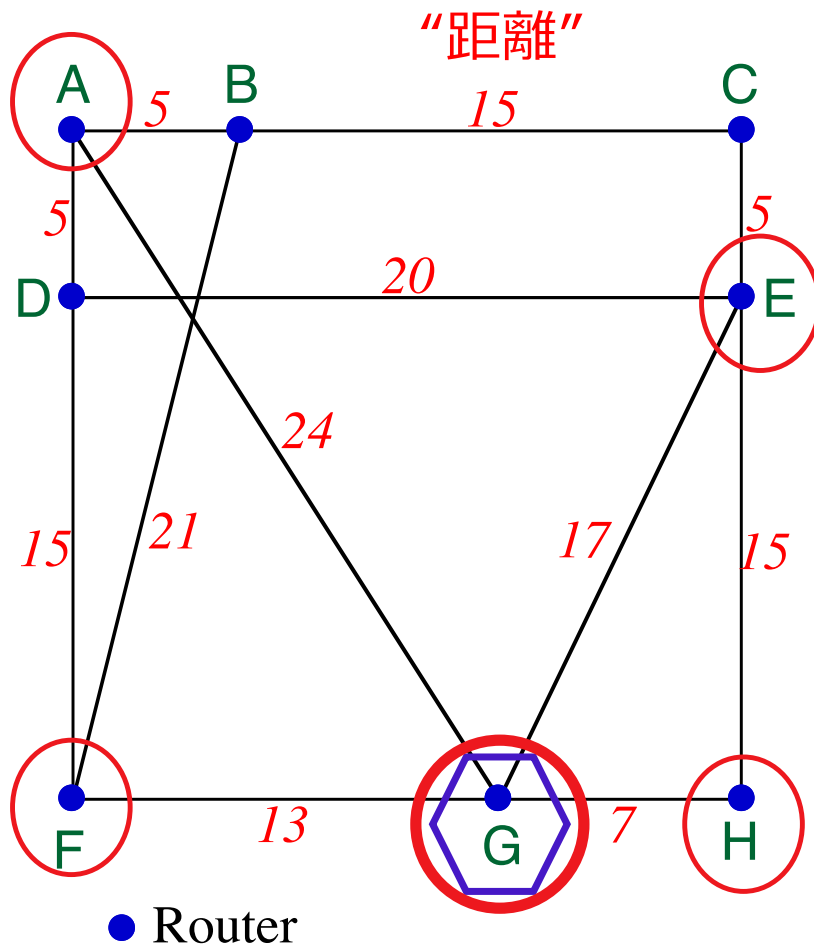
## ▶ リンク状態ルーティング

- ▶ 各ルータは**隣接ルータの情報**を**すべてのルータに**配布
- ▶ 得られた情報により、各ルータは全ネットワークトポロジーと**“距離”**を算出
- ▶ 例）**OSPF**, IS-IS

# 距離ベクトルルーティング

Gに隣接する4つのルータ  
から受信した距離ベクトル

Gの新しい  
経路表



To	A	E	F	H
A	0	25	20	31
B	5	20	21	35
C	20	5	35	20
D	5	20	15	35
E	25	0	30	15
F	20	30	0	20
G	24	17	13	7
H	31	15	20	0

G-A	G-E	G-F	G-H
24	17	13	7

Line	
24	A
29	A
22	E
28	F
17	E
13	F
0	-
7	H

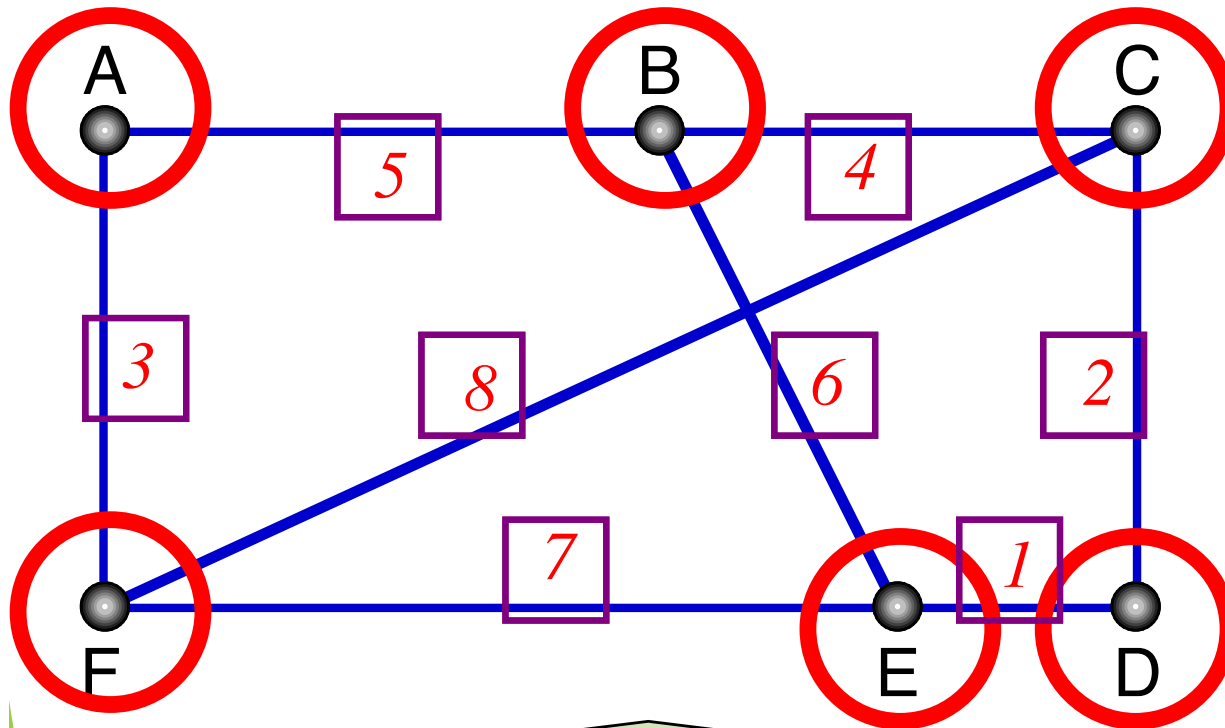
# リンク状態ルーティング

## 【各ルータでの処理】

1. すべての隣接ルータのアドレスの調査
2. 各隣接ルータへの“距離”（ホップ数や遅延など）を測定
3. **リンク状態パケット**の作成
  - ▶ 【含まれる情報】 自分のアドレス、シーケンス番号、年齢 (age)、すべての隣接ルータのアドレスとそこまでの“距離”
4. すべての他のルータへリンク状態パケットの配布
  - ▶ **Flooding**（周りにばらまく）による
5. すべての他のルータへの最短パスの計算



# リンク状態パケット



あるサブネット

リンク状態パケット

すべての他のルータに  
いかにリンク状態パケットを配布するか？

A	
Seq.	
Age	
B	5
F	3

B	
Seq.	
Age	
A	5
C	4
E	6

C	
Seq.	
Age	
B	4
D	2
F	8

D	
Seq.	
Age	
C	2
E	1

E	
Seq.	
Age	
B	6
D	1
F	7

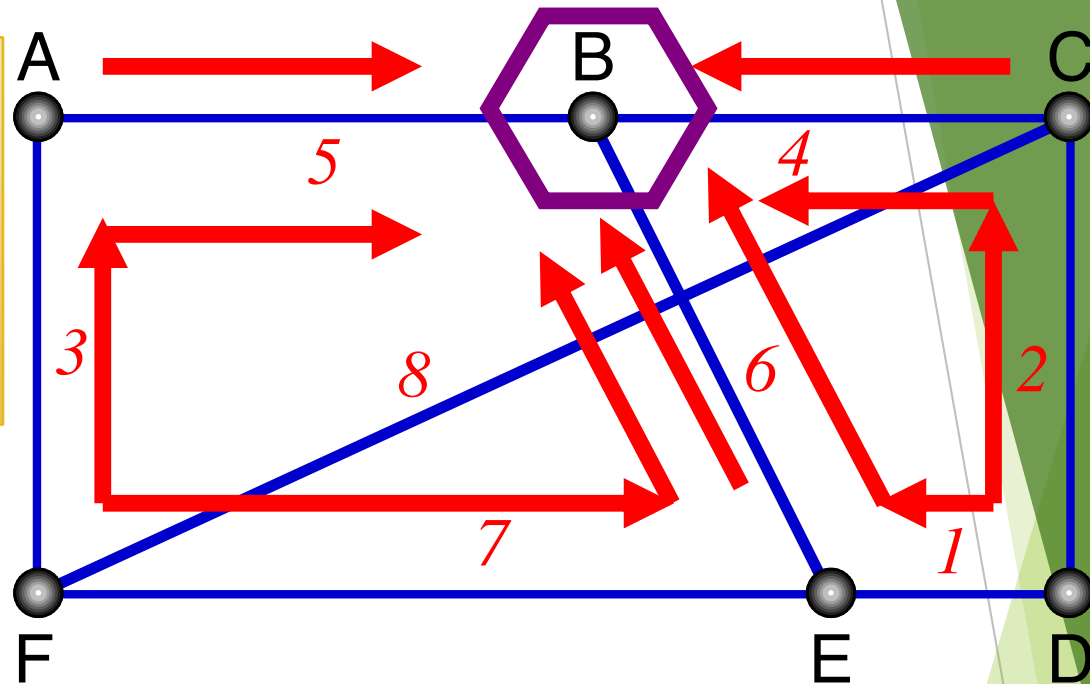
F	
Seq.	
Age	
A	3
C	8
E	7

**Flooding**を用いる

# Floodingによるリンク状態パケット (LSA) の配布

- 他から配布されたLSAは少しの間ルータのバッファ内で保持
- FloodingによるLSAの無駄な転送の抑制
- LSAに対するAckが必要

ルータBの  
パケット  
バッファ



送信フラグ 応答フラグ

送信元	順序番号	年齢	A	C	E	A	C	E	データ
A	51	60	0	1	1	1	0	0	
E	51	60	1	1	0	0	0	1	
F	51	59	0	1	0	1	0	1	
C	50	60	1	0	1	0	1	0	
D	51	59	1	0	0	0	1	1	

# その他のルーティング

- ▶ 階層的ルーティング

- ▶ すべてのルータはいくつかの領域(region)に分割される
- ▶ 各ルータは、自分の領域内のすべてのルータに対するルーティング情報を持つが、他の領域の内部構造の知識は持たない

- ▶ 移動端末へのルーティング

- ▶ 放送形ルーティング

- ▶ マルチキャストルーティング

# データ交換と ネットワーク層

TCP/IPにおけるルーティングプロトコル

※IPの詳細は次章のTCP/IPで

# インターネットのルーティング プロトコル

## AS※（またはドメイン）による階層的ルーティング

※AS（Autonomous System; 自律システム）  
統一された運用ポリシーによって管理されたネットワークの集合。  
BGPにより接続される単位となる

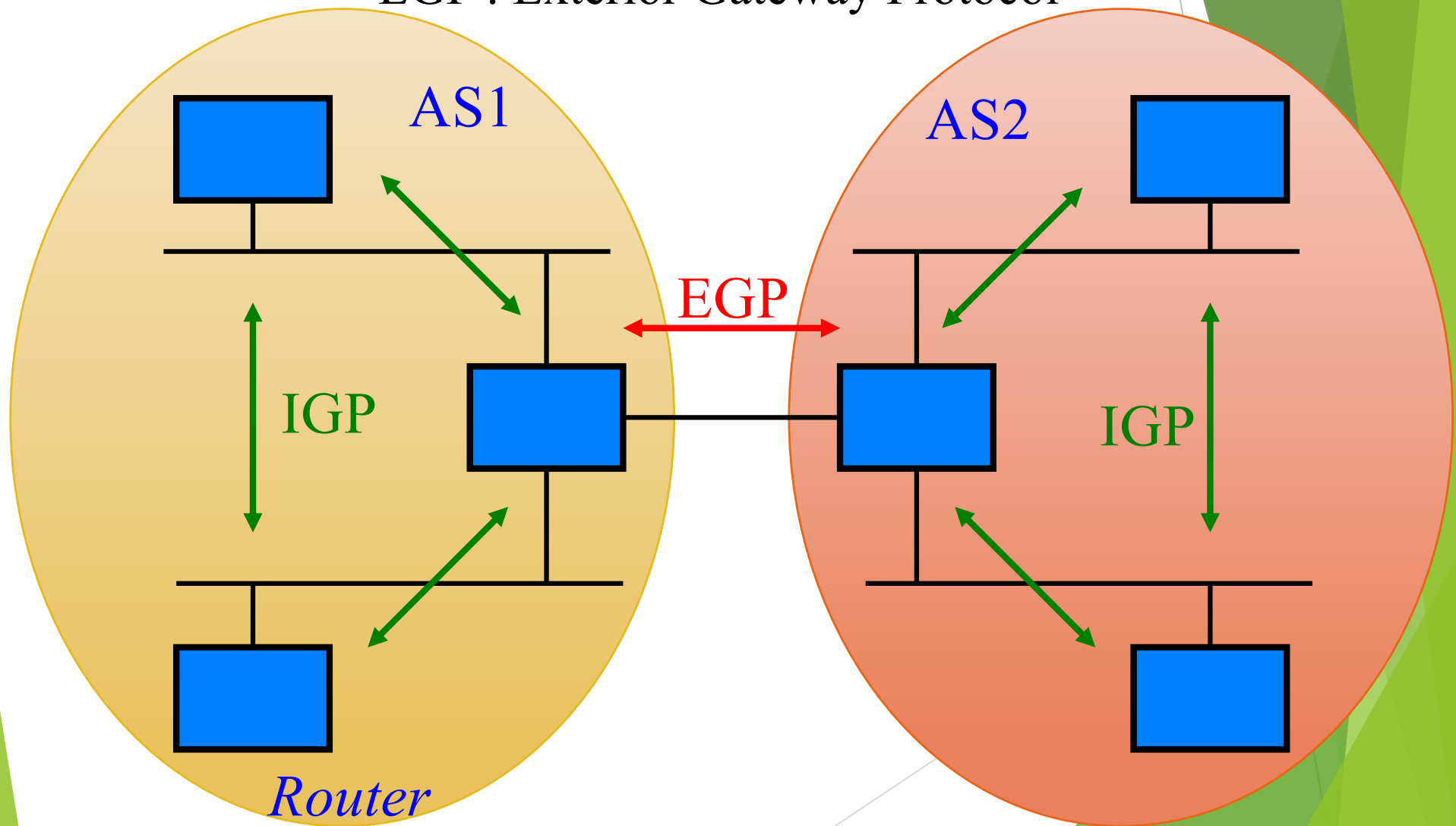
- ▶ AS内ルーティング
  - ▶ RIP(Routing Information Protocol)
  - ▶ IGRP(Interior Gateway Routing Protocol)
  - ▶ OSPF(Open Shortest Path Fast)
  - ▶ IS-IS
- ▶ AS間ルーティング
  - ▶ EGP(Exterior Gateway Protocol)
  - ▶ BGP(Border Gateway Protocol) またはIDRP(Inter-Domain Routing Protocol)

# IGPとEGP

EGPという名前のEGPもある ☺

IGP : Interior Gateway Protocol

EGP : Exterior Gateway Protocol



# データ交換と ネットワーク層

TCP/IPにおける距離ベクトルルーティング

# TCP/IPにおける距離ベクトル型ルーティングプロトコル

- ▶ RIP (Routing Information Protocol)
  - ▶ RFC 1058 (HistoricなProtocol)
  - ▶ 動的なIGPの一つ
  - ▶ Unixのroutedとして広く実装
  - ▶ ネットワークの距離をMetricで表
    - ▶ 小さいMetric程良い経路
    - ▶ Metricは1 ~ 15  
(実装により16を無限大とする)



# RIPの特徴

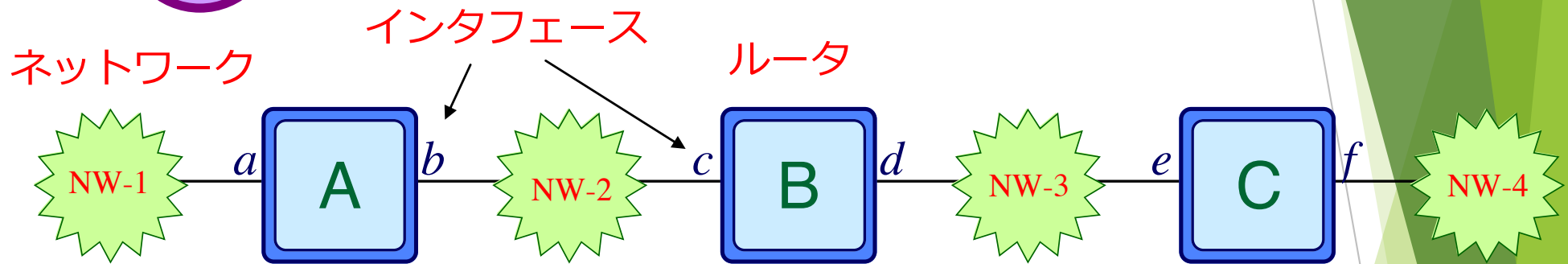
## ▶ RIPの特徴

- ▶ Metricをホップ数(Hop count)  
(ルータを経由する回数) で定義
- ▶ 送信先へは一つのルート
- ▶ 30秒毎のFlooding  
180秒更新されない経路の  
Metricを16とし, 更に120秒後に消去
- ▶ サブネットに対応していない
  - ▶ RIP version 2で対応

# RIPの動作

1

自分と**直接接続**している  
ネットワークへの経路を作成



経路表 (A)		
NW-1	a	0
NW-2	b	0

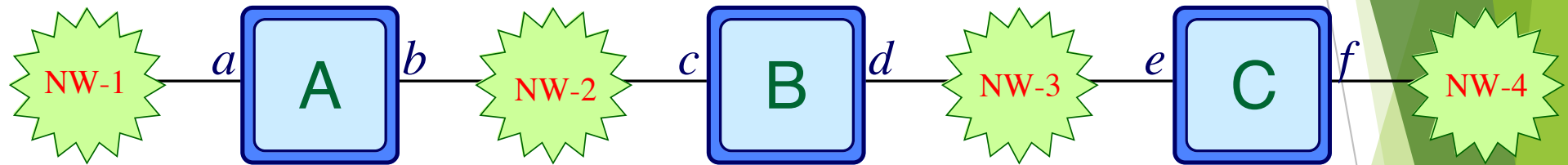
経路表 (B)		
NW-2	c	0
NW-3	d	0

経路表 (C)		
NW-3	e	0
NW-4	f	0

# RIPの動作

2

自分と隣接するルータと  
経路情報（距離ベクトル）を交換



経路表 (A)		
NW-1	a	0
NW-2	b	0

NW-2	b	1
NW-3	b	1

経路表 (B)		
NW-2	c	0
NW-3	d	0

NW-1	c	1
NW-2	c	1
NW-3	d	1
NW-4	d	1

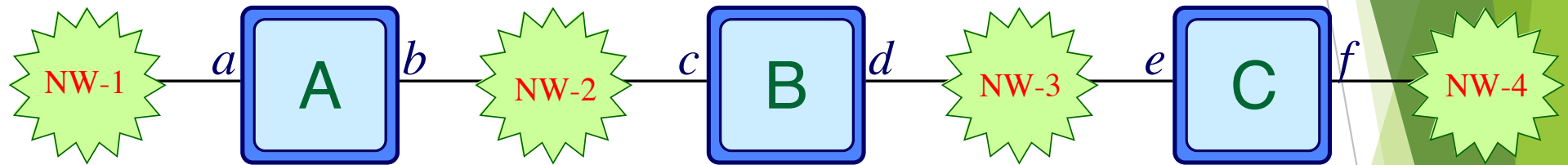
経路表 (C)		
NW-3	e	0
NW-4	f	0

NW-2	e	1
NW-3	e	1

# RIPの動作

3

不要な経路情報を削除  
(距離の小さいものを選択)



経路表 (A)		
NW-1	a	0
NW-2	b	0
NW-2	b	1
NW-3	b	1

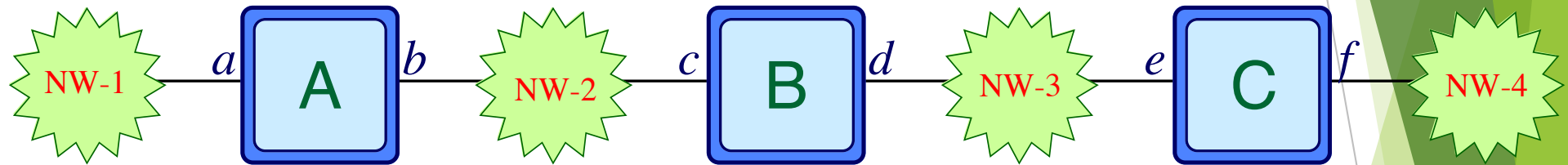
経路表 (B)		
NW-2	c	0
NW-3	d	0
NW-1	c	1
NW-2	c	1
NW-3	d	1
NW-4	d	1

経路表 (C)		
NW-3	e	0
NW-4	f	0
NW-2	e	1
NW-3	e	1

# RIPの動作

4

経路表を更新し,  
新たに経路情報を交換



経路表 (A)		
NW-1	a	0
NW-2	b	0
NW-3	b	1

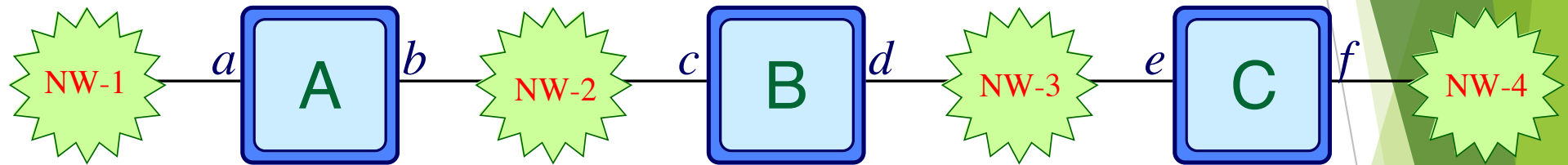
経路表 (B)		
NW-2	c	0
NW-3	d	0
NW-1	c	1
NW-4	d	1

経路表 (C)		
NW-3	e	0
NW-4	f	0
NW-2	e	1

# RIPの動作

5

最終的な経路表の完成  
(以下これを定期的に繰り返す)



経路表 (A)		
NW-1	a	0
NW-2	b	0
NW-3	b	1
NW-4	b	2

経路表 (B)		
NW-2	c	0
NW-3	d	0
NW-1	c	1
NW-4	d	1

経路表 (C)		
NW-3	e	0
NW-4	f	0
NW-2	e	1
NW-1	e	2

# データ交換と ネットワーク層

TCP/IPにおけるリンク状態ルーティング

# TCP/IPにおけるリンク状態型 ルーティングプロトコル

## ▶ OSPF(Open Shortest Path Fast)

- ▶ RFC2328 (Apr. 1998) ※TCP/IPの標準文書

- ▶ 240ページ超 ※RIPv1は33ページ

- ▶ 動的なIGPの一つ

- ▶ 直接IPでカプセル化 ※トランスポート層を使わない

- ▶ プロトコル番号89 ※RIPv1はUDPを使う

- ▶ 精密な経路制御, 経路の素早い収束

- ▶ UNIXでは, gatedとして実装

- ▶ ネットワークの距離はコストで定義

- ▶ サブネットマスクに対応



# コスト(Cost)

- ▶ ルータ間の距離指標
- ▶ 1から65535までの値を取る
- ▶ 対地までの全リンクのコスト総和を比較して経路を選択
- ▶ コストが等しい複数の径路が存在する場合は、負荷分散が行われる

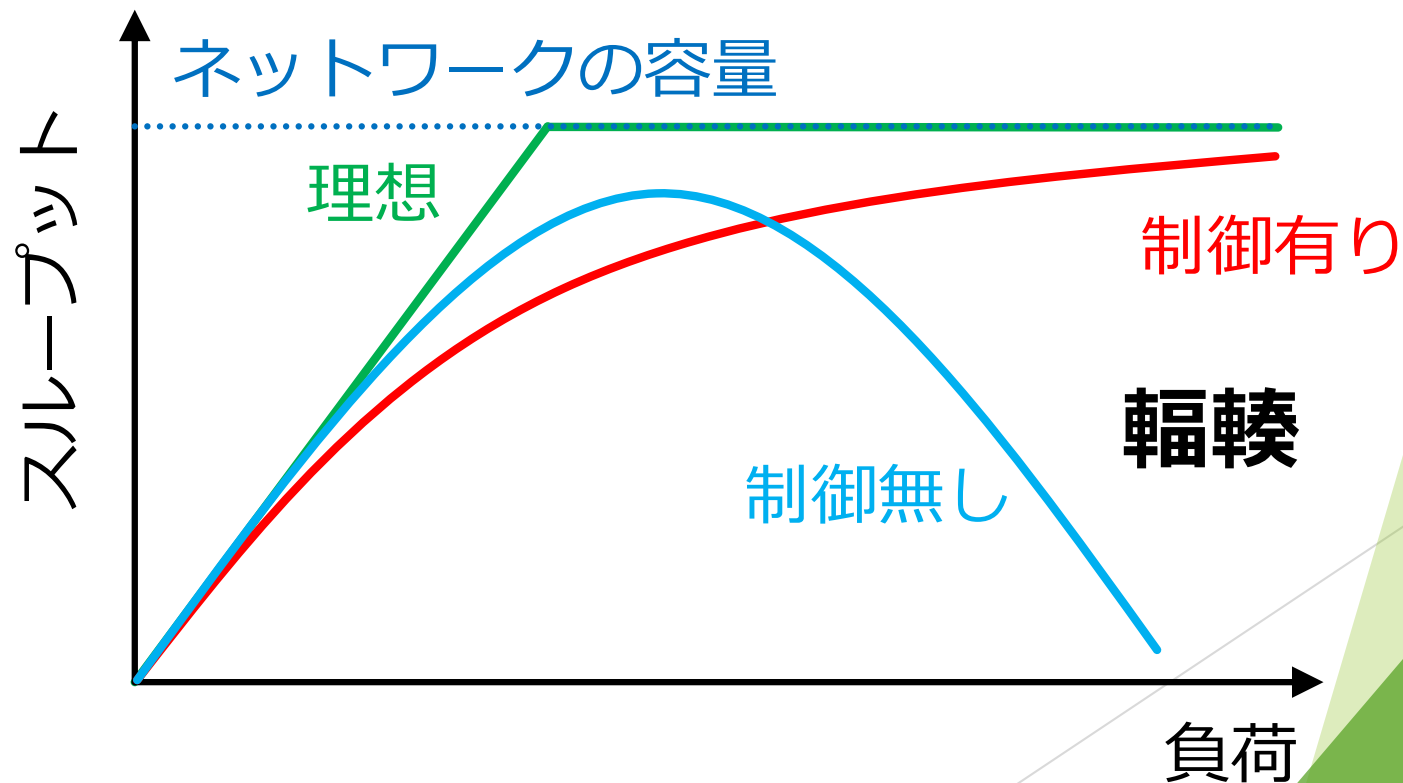
# データ交換と ネットワーク層

輻輳制御

※ ネットワーク層以外でも行われる

# 輻輳制御

- ▶ 輻輳 : ネットワーク負荷の増加による性能の低下
  - ▶ 原因 1 : ルータや端末の処理速度が遅い
  - ▶ 原因 2 : 通信回線の容量を越えた入力トラヒック



# 輻輳制御方法

## ▶ 呼受付制御

輻輳が生じたら新たに発生したパケットをネットワークに入れない（コネクション型パケット交換）

## ▶ フロー制御

※トランスポート層

【例】 TCPウィンドウフロー制御, スロースタート

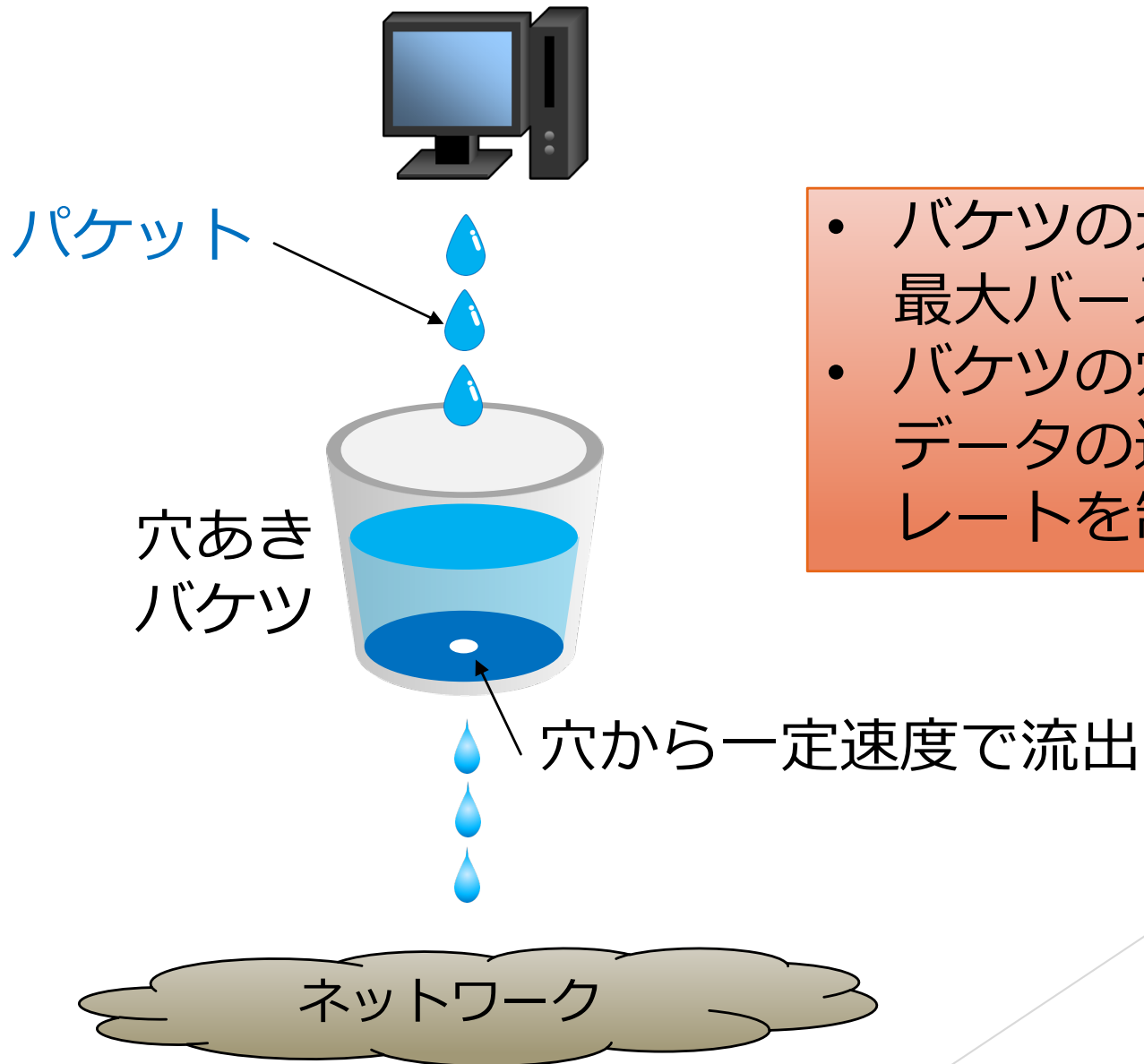
## ▶ トラフィックシェーピング（ルータでの処理）

【例】 リーキバケットアルゴリズム, トークンバケットアルゴリズム

## ▶ パケット廃棄（ルータでの処理）

【例】 RED(Random Early Detection)

# リーキバケットアルゴリズム (Leaky bucket; 穴あきバケツ)



- バケツの大きさが最大バースト長を制限
- バケツの穴から漏れるデータの速度がピークレートを制限

# トークンバケットアルゴリズム

1秒毎に  $r$  個の  
トークンが加えられる

Token arrival rate  $r$

トークンバケット

- バケツは最大  $b$  個の  
トークンを蓄えられる
- 溢れたトークンは廃棄

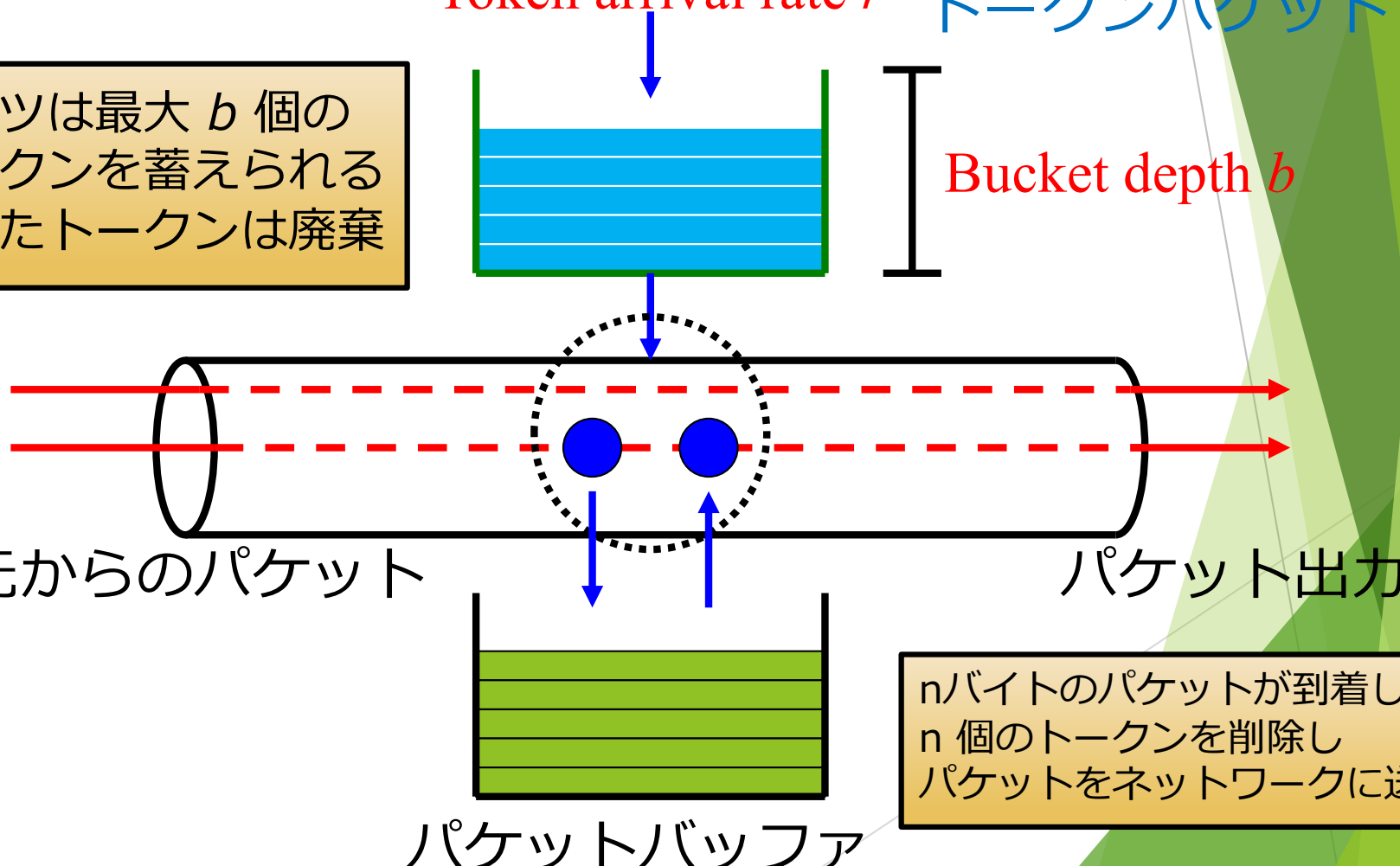
Bucket depth  $b$

送信元からのパケット

パケット出力

パケットバッファ

$n$ バイトのパケットが到着したら  
 $n$  個のトークンを削除し  
パケットをネットワークに送出



# RED(Random Early Detection)の動作

ルータでバッファのオーバフローが生じる前にパケットを廃棄

