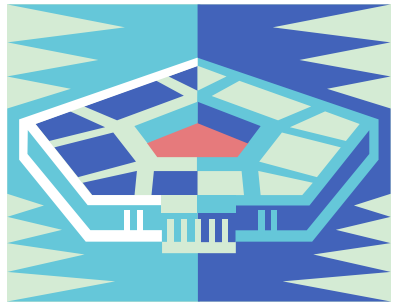
インターネットとは？

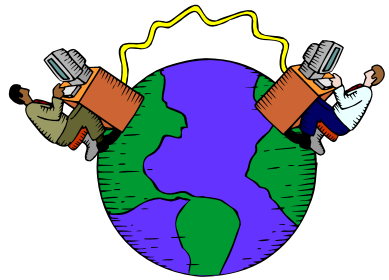
- “**inter**-”と “**net**work” の造語
  - 複数のネットワークをつなぐ
- “The Internet”
  - LAN (local-area network) をルータ (router)でつなぐことで構成された世界規模の広域ネットワーク
- 1969年 DARPA（米国国防総省高等研究プロジェクト）として構築されたARPANETがルーツ
  - 全米主要大学を中心に発展

インターネットの特徴

- 自律分散型であること
  - 動的な経路制御
    - 障害時にも人手を介することなく自動的に迂回
  - すべての構成要素が論理的に対等
    - どこに障害が起きても他への影響が最小限
- コンピュータ指向であること
  - 途中のネットワークは可能な限りシンプルに
  - 複雑な制御は両端のコンピュータに任せる
    「End-to-end 原理」
- 大学での研究開発がベース
  - 分散アルゴリズムの集大成
  - UNIXとともに発展



プロトコルって？



- プロトコル(protocol)
  - 原義：外交上の約束
  - （ネットワーク用語）
    - 異なるシステムの間で制御やデータのやりとりに関してあらかじめ定めた約束事
- インターネットのプロトコル：TCP/IP
  - Internet Protocol (IP) を含む多くのプロトコルの総称
  - IETF (Internet Engineering Task Force) にて標準化、RFCとして公開

郵便システムにおける「プロトコル」

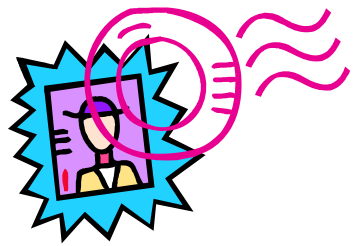
（東京のあなた）

1. はがきを用意
   - 官製はがき、または私製はがき（所定のサイズ）
2. 宛名を記入
   - 郵便番号、住所、氏名
3. 50円切手

   （私製はがきのみ）
4. 郵便ポストに投函

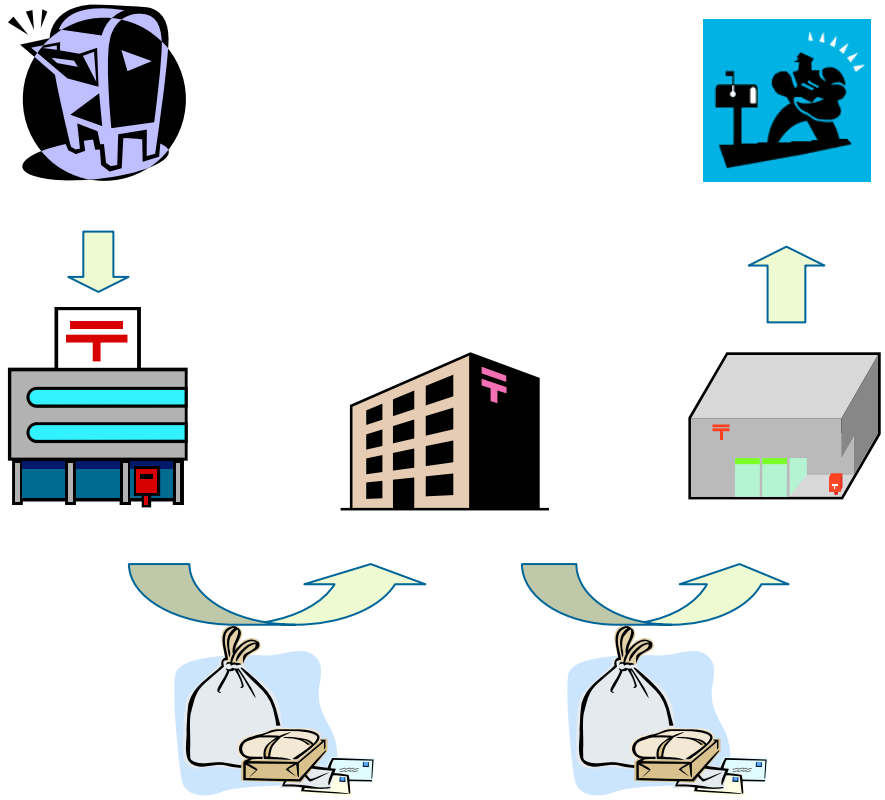
（京都の友人）

1. 郵便受を用意
2. 郵便受を毎日チェック
3. はがきが届いていれば取り出す



郵便システムにおける「プロトコル」の階層

- 集配局の手順
  - １日何回かの集配
  - 郵便番号を用いて仕分
  - 次の目的局別配送袋につめて
  - 発送
- 郵便局間の配送手順
  （鉄道の場合）
  - どの列車のどの車両に
  - 何時にどこの駅で誰が…



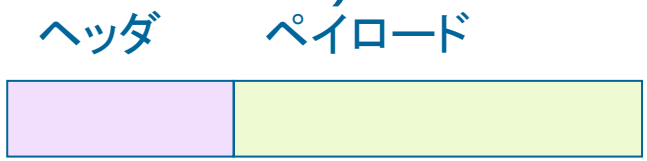
内部的に手順が階層化され、各手順では自階層以外のことは考慮しなくてよい

TCP/IP におけるプロトコルの階層

ARPA モデル				OSI参照モデル
プロセス／アプリケーション	HTTP（Web），SMTP（メール），FTP		NFS SNMP	アプリケーション層 プリゼンテーション層 セッション層
ホストto ホスト	TCP	RTP UDP		トランスポート層
インターネット	IPv6	ICMPv6		ネットワーク層
ネットワークインターフェース	Ethernet（IEEE802.3），無線LAN（IEEE802.11b），IEEE1394			データリンク層
	伝送媒体：UTP，同軸ケーブル，光ファイバ，無線媒体など			物理層

Internet Protocol (IP)

- パケット通信の一種
  - すべてのデータを可変長のパケット(packet)に分割して伝送
  - パケット（packet:小包）
    - ヘッダ（header:荷札）とペイロード（payload:荷物）の組
- IPのパケット（IP datagram）
  - ヘッダ
    - 送信元アドレス(source address)
    - 宛先アドレス(destination address)
    - ペイロード長

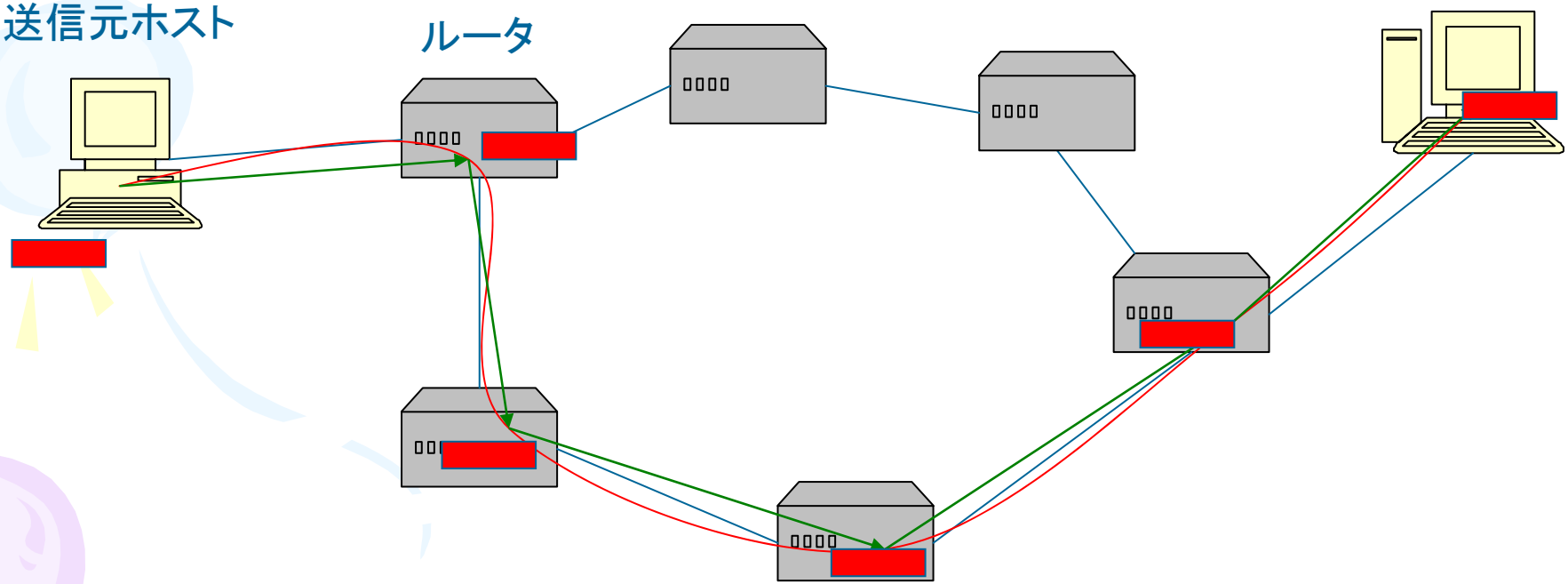


IPアドレス

- 現行のIP (IPv4)
  - 全世界で一意性の保証された32bit整数値
    - 2^32＝約40億
  - ネットマスクにより以下に分離
    - ネットワークアドレス部（上位）
      - ネットワークを全世界で識別
    - ホストアドレス部（下位）
      - ネットワーク内でホストを識別
- 次世代IP (IPv6)
  - アドレス空間を128bitに拡張
    - ネットワークアドレス部64bit，ホストアドレス部64bit

インターネットにおける パケットの配送

IPヘッダに書かれた宛先ホストのIPアドレス
のみから次ホップルータを決定する



経路制御 (routing)

経路制御の考え方(1)
階層的経路制御

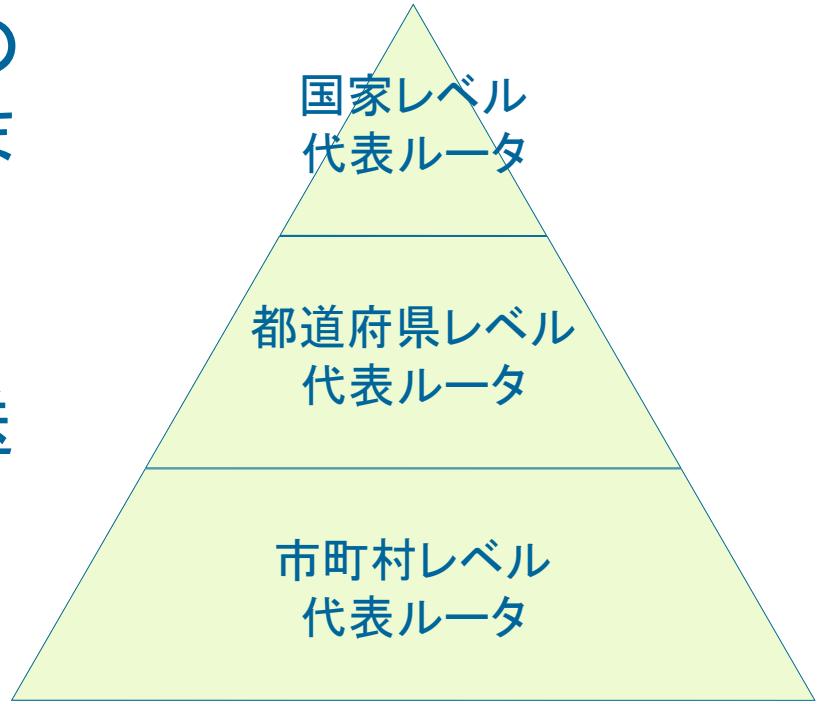
階層的経路制御のアルゴリズム

- 宛先がその階層で同一のエリアであれば、そのまま配送
- 他のエリアであれば上位階層の代表ルータへ転送

（欠点）

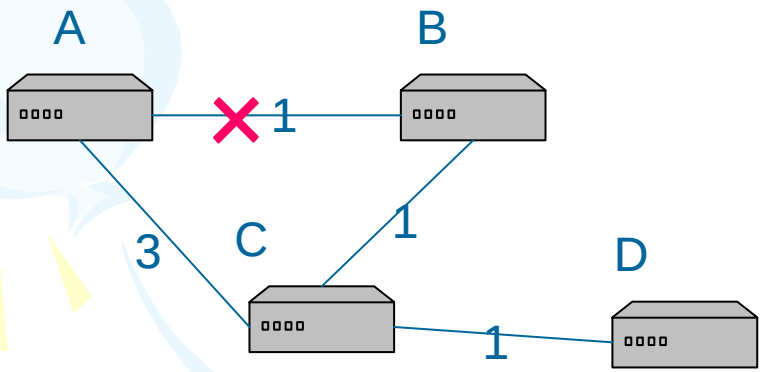
- 代表ルータが『弱点』

  single point of failure



経路制御の考え方(2)
静的経路制御と動的経路制御

- 経路表 (routing table)
  - 各ルータが、宛先アドレスと次ホップルータの組(経路エントリ)を表にして保持する



ルータAにおける経路表

宛先	距離	次ホップ
B	1	B
C	2	B
D	3	B

- 静的経路制御
  - 経路表をあらかじめ計算しておく

  ⇒障害時の迂回が自動ではできない
- 動的経路制御
  - ルータが自律分散的なアルゴリズムに従い経路表を動的に計算

リンクA－B障害時

宛先	距離	次ホップ
B	4	C
C	3	C
D	4	C

経路制御プロトコル (routing protocol)

- 経路制御アルゴリズム
  - 各ルータで経路表を構成するための分散アルゴリズム
  - 代表となる特定のルータを仮定せず、自律分散的に動作する
  - ネットワークの状況に応じて経路表を動的に再構成
- 経路制御プロトコル
  - 経路制御アルゴリズムにおいてルータ間での情報のやり取りを定めた手順
    - 距離ベクトル型 (RIP)
    - パスベクトル型 (BGP)
    - リンク状態型 (OSPF, P-NNI, IS-IS)

距離ベクトル型経路制御アルゴリズム

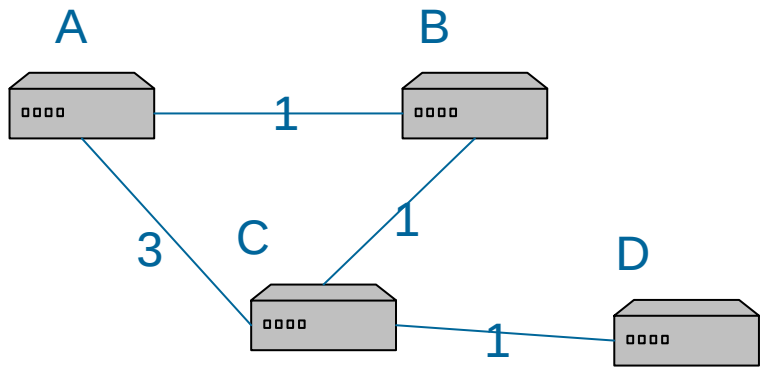
1. 初期状態

   各ルータは自分に直下のホストに関する情報のみを持つ

2. 隣接ルータと経路表を交換
   - 自分の経路表にない宛先
   - 自分の持っている経路より近い経路

   に関する情報があれば、自分の経路表を更新

3. 以上を繰り返すことで、すべてのルータの経路表が最短経路に従う



宛先	距離	次ホップ
B	1	B
C	3	C

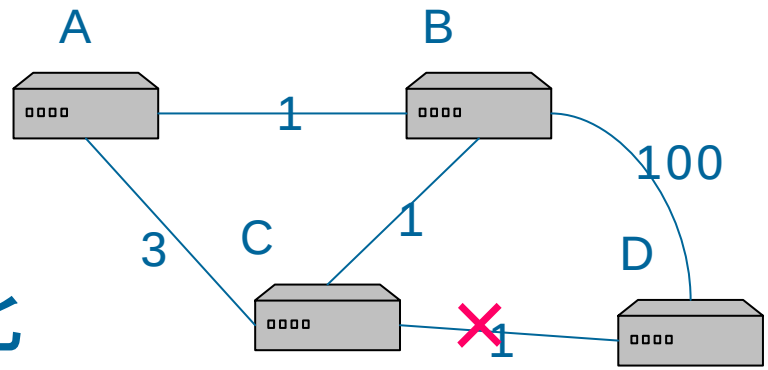
宛先	距離	次ホップ
B	1	B
C	2	B
D	4	C

宛先	距離	次ホップ
B	1	B
C	2	B
D	3	B

距離ベクトル型アルゴリズム
「無限のカウント」の問題

C-Dリンク切断時

A, B, C のDに対する
経路エントリの時刻変化



時刻	(A)	(B)	(C)	備考
-	D 3 B	D 2 C	D 1 D	初期状態
0	D 3 B	D 2 C	D ∞ -	リンク断
1	D 3 B	D ∞ -	D 6 A	
2	D ∞ -	D 7 C	D 6 A	
3	D 8 B	D 7 C	D ∞ -	

Good news travels fast, ill news travels slowly.

実際の経路制御(1)

IGP（組織内経路制御）

⇒ リンク状態型

（考え方）

- 経路表ではなくリンク状態（ルータ間の接続関係そのもの）をルータ間で交換する
- 全ルータがすべての接続トポロジーを知っていれば、各ルータで最短経路木(shortest path tree)をDijkstraのアルゴリズムにより計算できる
- リンク状態が変化した場合には差分のみを伝達
  - 障害／復旧時には即座に再計算

（欠点） ルータでの計算負荷が高い

（例） OSPF

実際の経路制御(2)

EGP（組織間経路制御）

⇒　パスベクトル型

（考え方）

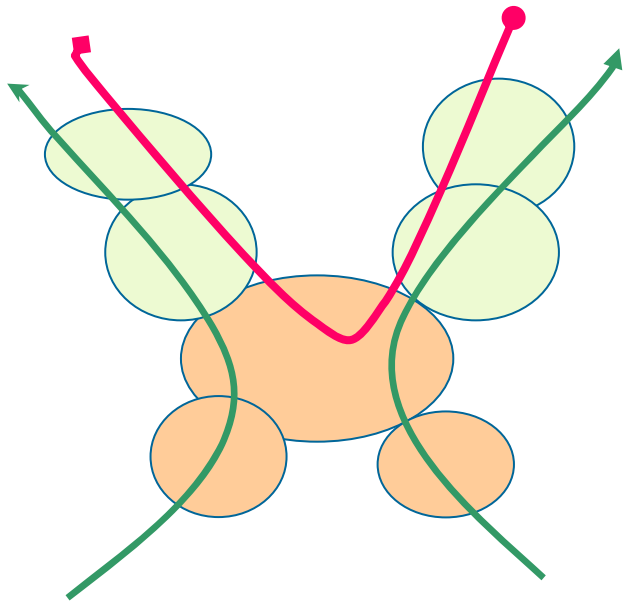
- 組織(AS; Autonomous System)間の経路制御では距離だけでなくポリシーを考慮した経路制御が必要

  (例) 文部科学省のSINETは、ISP間のトラフィックを通過(transit)させない

（欠点）

– 経路エントリ数が増大

（例）BGP



TCP とUDP

- TCP/IPにおけるトランスポート層プロトコル
  - ホスト間のend-to-end通信を規定
  - 「データ」をIPパケットに分割して送るための方法
- TCP (Transmission Control Protocol)
  - 信頼性のあり
  - 一対一
- UDP (User Datagram Protocol)
  - 信頼性なし
  - 一対多も可

信頼性のあるデータ伝送とは？

- データの欠落があると困る場合
  - 電子メールなどの文書
  - 実行可能プログラム
  - 静止画像

⇒TCP

- 遅延が一定値以下であるほうが重要な場合
  - リアルタイムマルチメディアデータのストリーム配信
    - データが欠落した場合、再送するよりも、送れなかったデータを破棄して最新のデータを送る方がbetter

⇒UDP

TCPによる信頼性のあるデータ伝送の原理

- IPパケットのサイズの制限
  - Ethernet系の場合、1500バイト／パケット

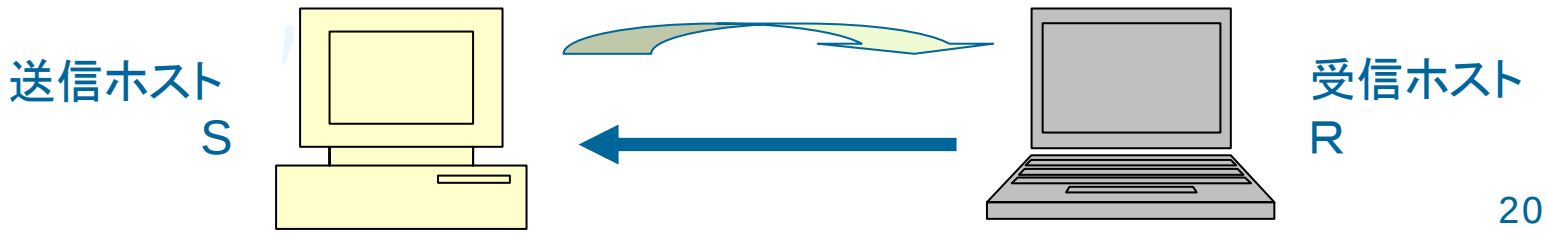
⇒大きなデータは複数のパケットに分割して順次伝送

素朴なアルゴリズム

信頼性のない伝送路(IP)を用いて信頼性のあるデータ伝送を行う

S：送信ホスト、R：受信ホスト

- SからRに、パケットを１個送る。
- Rはパケットを受け取ったら、そのことを伝える応答(肯定的応答)を送る。
- Sは、肯定的応答を受け取ったら次のパケットを送る
- Sは、パケットを送ってから一定時間待って応答が来なかったら、パケットが到達せずに失われたと判断し、再送する。



素朴なアルゴリズムの問題点

- データの伝送速度が送信ホストと受信ホストの間をパケットが往復する時間(Round Trip Time; RTT)に律速される
  - 送信されるパケットは高々RTTごとに1個

（試算）パケットサイズの最大値が1500バイト、RTTが10ミリ秒のとき

1500バイト/0.01秒、すなわち1.2Mbpsが限界

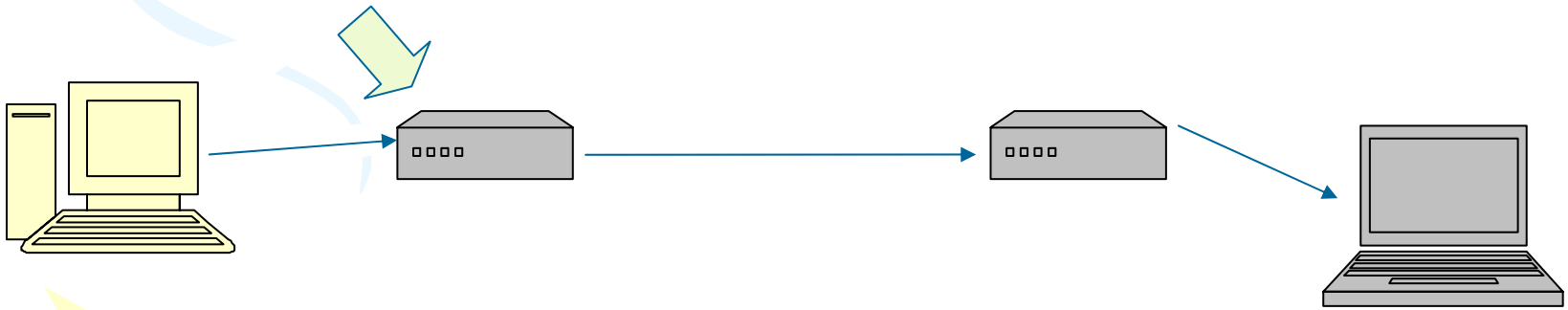
⇒パケットに順に番号(sequence number)を振り、一度にまとめて送り、応答(ACK)には正しく受信できたパケットの番号の最大値を添える

たくさんまとめて送るといくらでも速くなりますか？

- NO!
  - 受信側プロセスの能力
  - 伝送路途中の律速ルータ

  処理できないパケットは破棄

送信者はいかにして最適な送信レートを知るのか？



TCPにおける輻輳制御

- 輻輳ウィンドウによる管理
  - 送信側で管理される、一度にまとめて送ることのできるデータのサイズ
  - 送信開始時は1パケット分
  - 肯定的応答(ACK)の受信によりパケットの送達が確認されたら、サイズを2倍にする
    - ただし上限あり
  - 重複ACK（すなわちパケットの抜け）やタイムアウトが観測されたら小さくする

TCPの輻輳制御アルゴリズム

- TCP Tahoe
  - 4.3BSD Tahoe版の実装
- TCP Reno
  - 4.3BSD Reno版の実装；長らく標準
- TCP NewReno
  - 現在の標準

今なお活発に研究中

公平性(fairness)に配慮が必要