画像情報特論 (6)

- アダプテーション (2)

パケット廃棄対策、TCPフレンドリ

情報理工学専攻　甲藤二郎

E-Mail: katto@waseda.jp

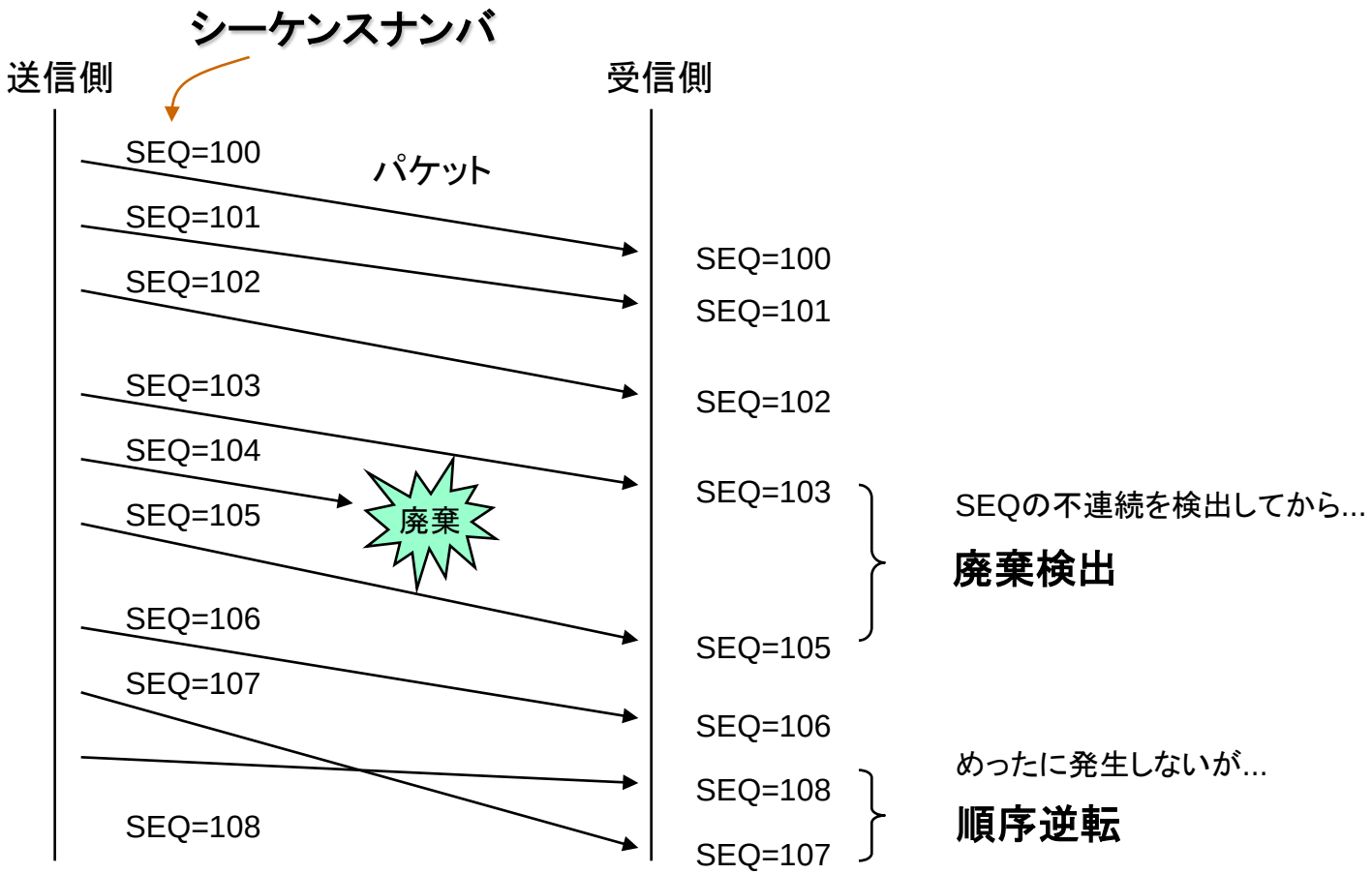
パケット廃棄対策

誤り対策一覧

	電話	移動体	ディジタル放送	インターネット	程度
誤り検出符号	○	○	○	○ (TCP/UDP)	検出 (ビット誤り)
シーケンス ナンバ		○	○	○ (RTP)	検出 (パケット廃棄)
再同期	△	○	○	○ (RTP)	局所化
インタリーブ	△	○	○		訂正
FEC				○ (RFC2733)	訂正
再送				△ (検討中)	訂正
NewPred	△	△	△	△ (検討中)	局所化

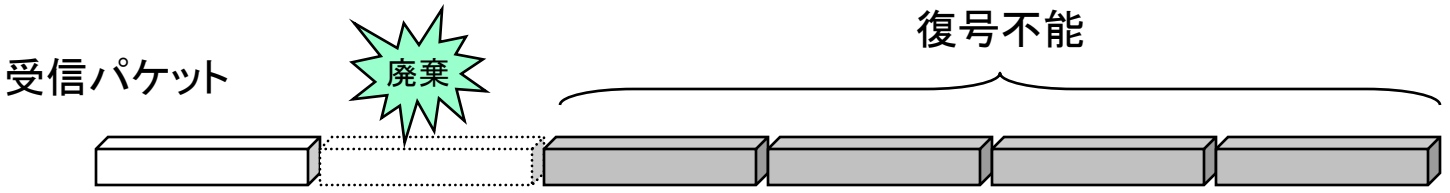
シーケンスナンバ

- パケット廃棄の「検出」

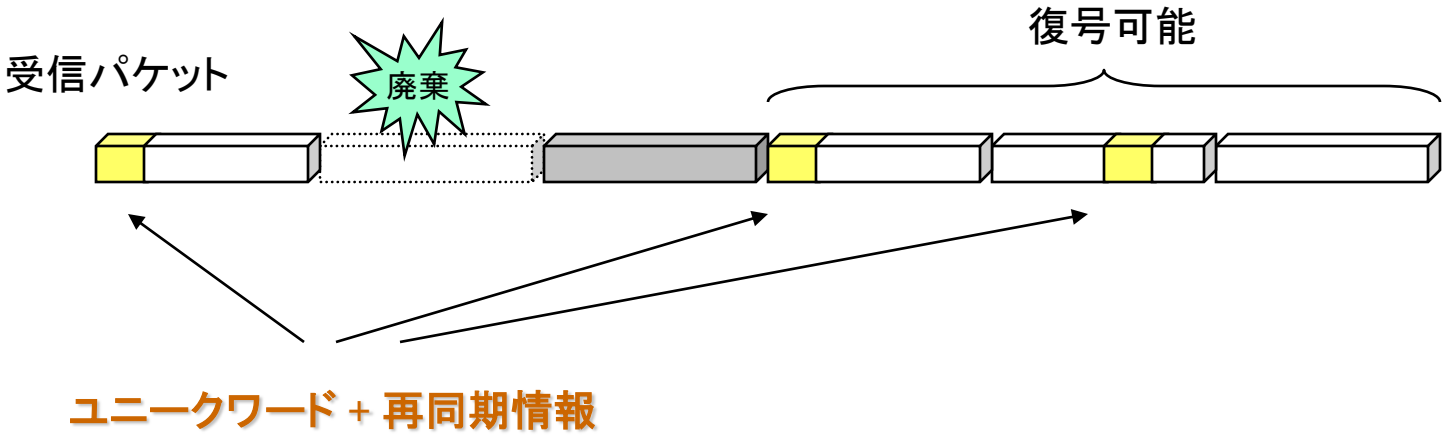


再同期

- パケット廃棄の影響の「局所化」



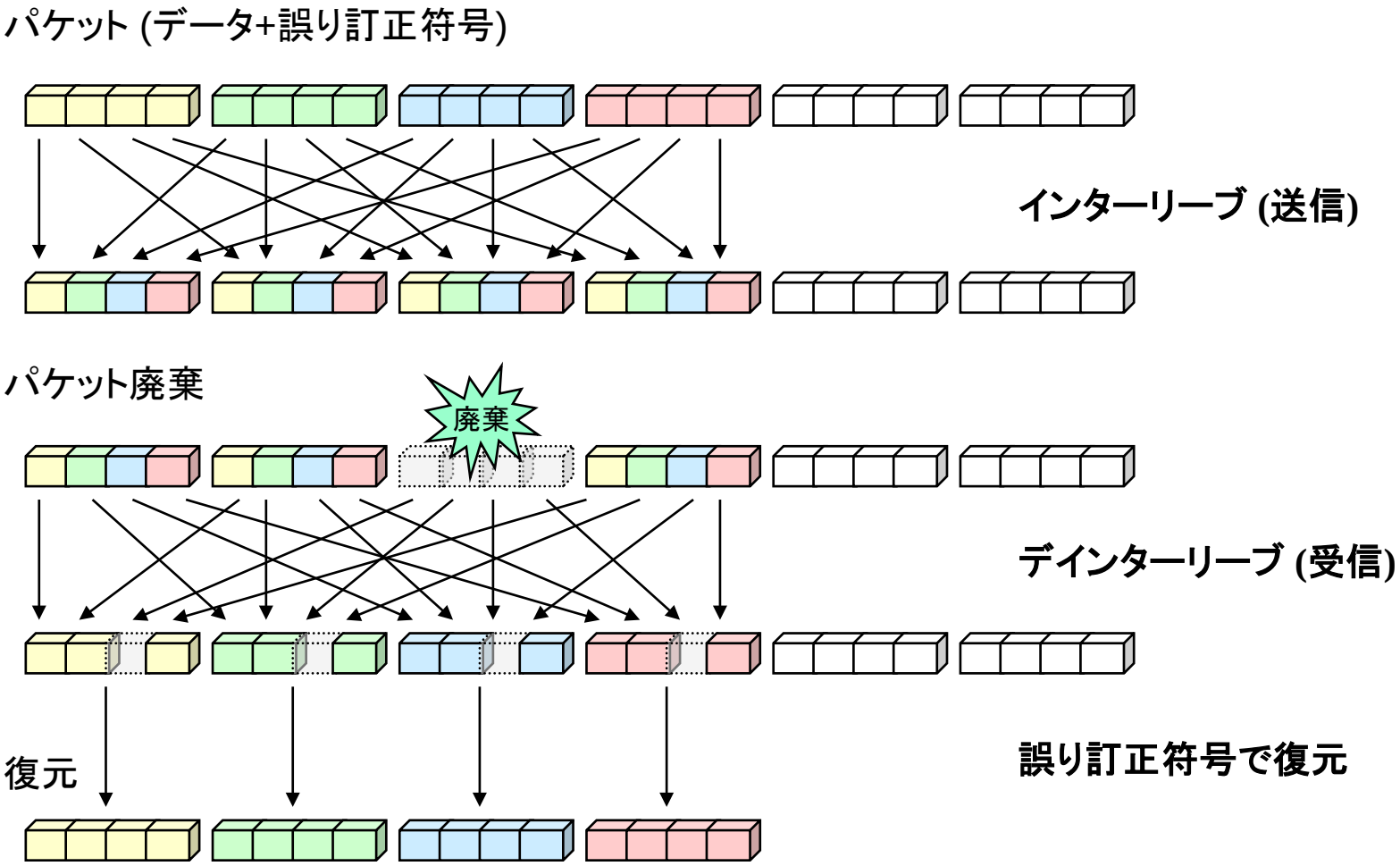
(a) 再同期情報がない場合



(b) 再同期情報がある場合

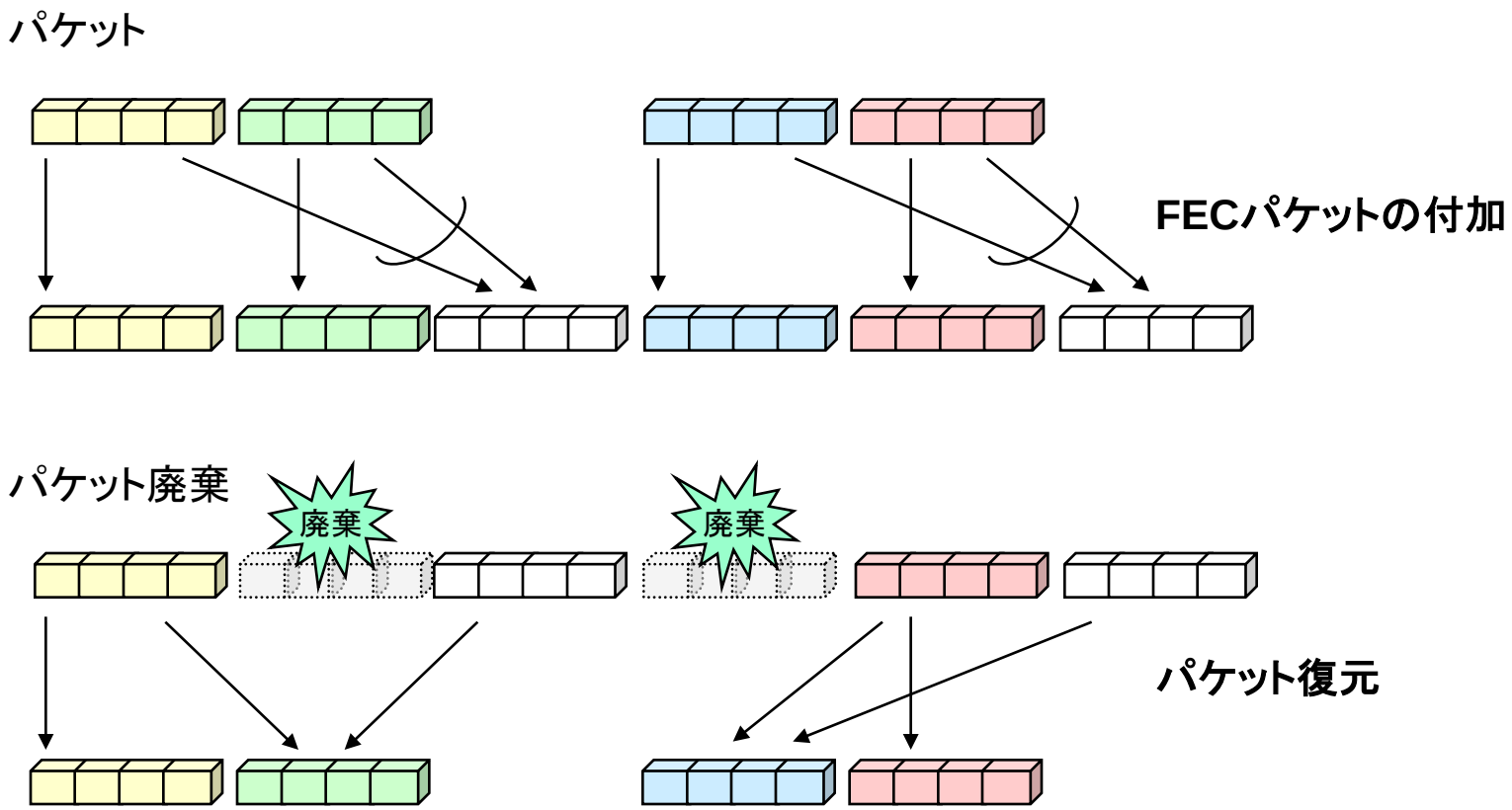
インタリーブ

- パケット廃棄の「訂正」… 誤り訂正符号の応用 1



FECパケット

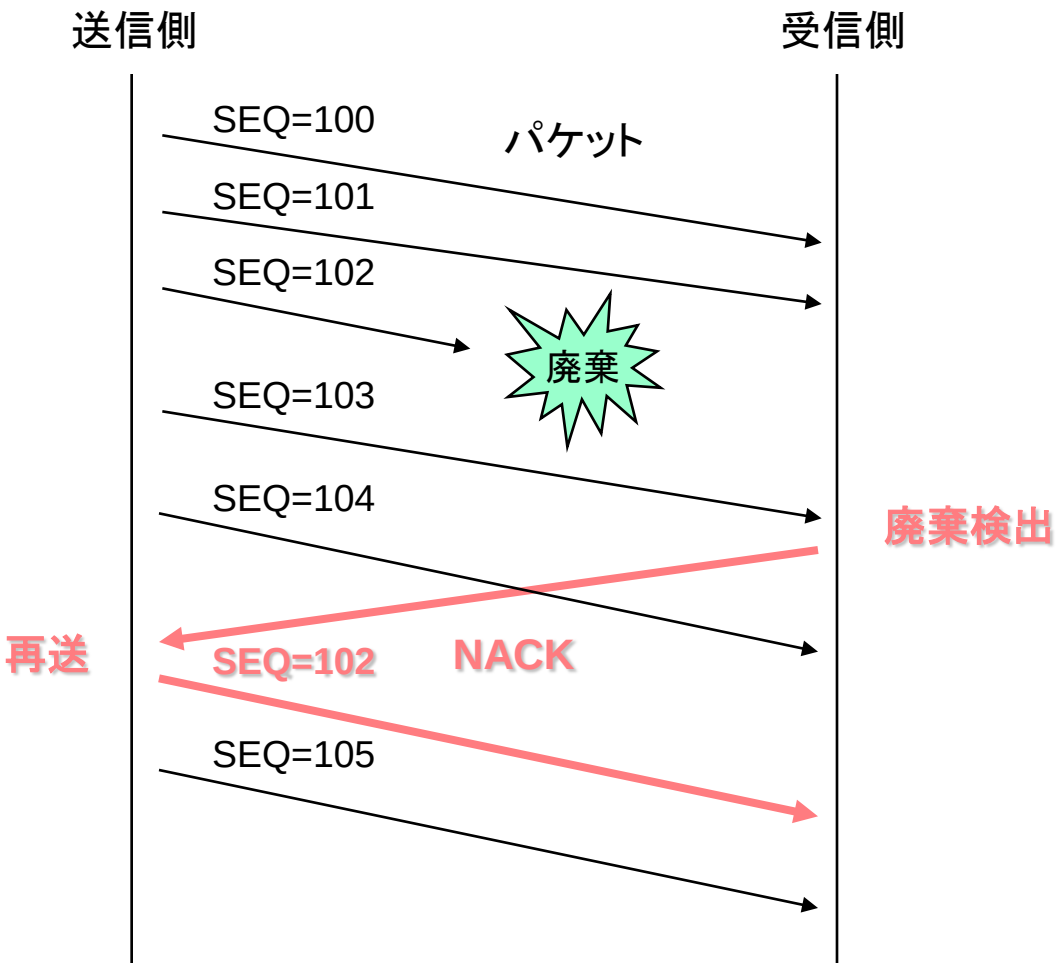
- パケット廃棄の「訂正」… 誤り訂正符号の応用 2



- パリティチェック
- Digital Fountain (トルネード符号)

再送

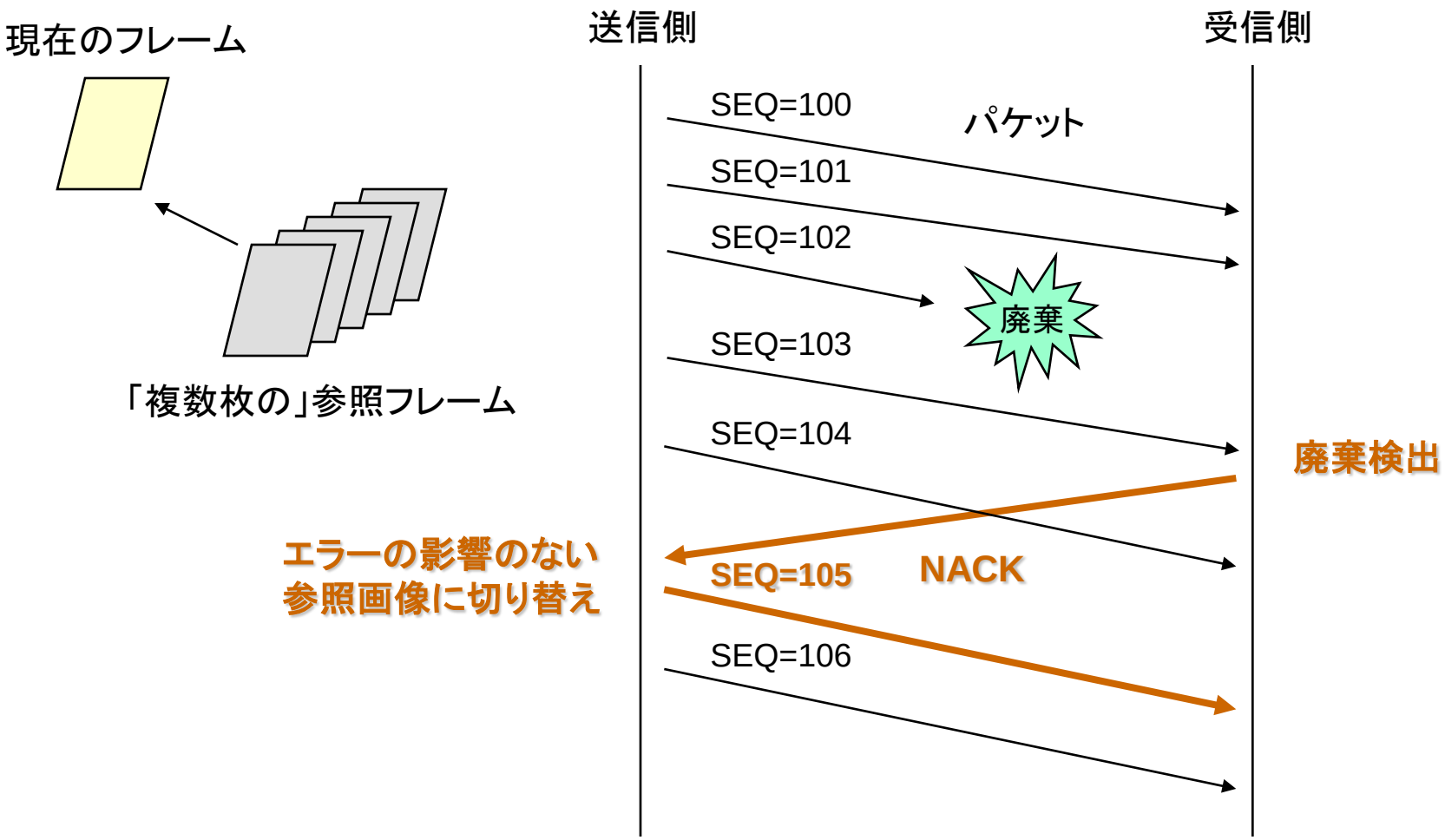
- NACK と廃棄パケットの「再送」



再送遅延が問題

NewPred

- NACK と「参照フレームの切り替え」



RTPペイロードフォーマット
(再同期)

RTPヘッダ

パケットタイプ: **転送メディアの符号化アルゴリズム**

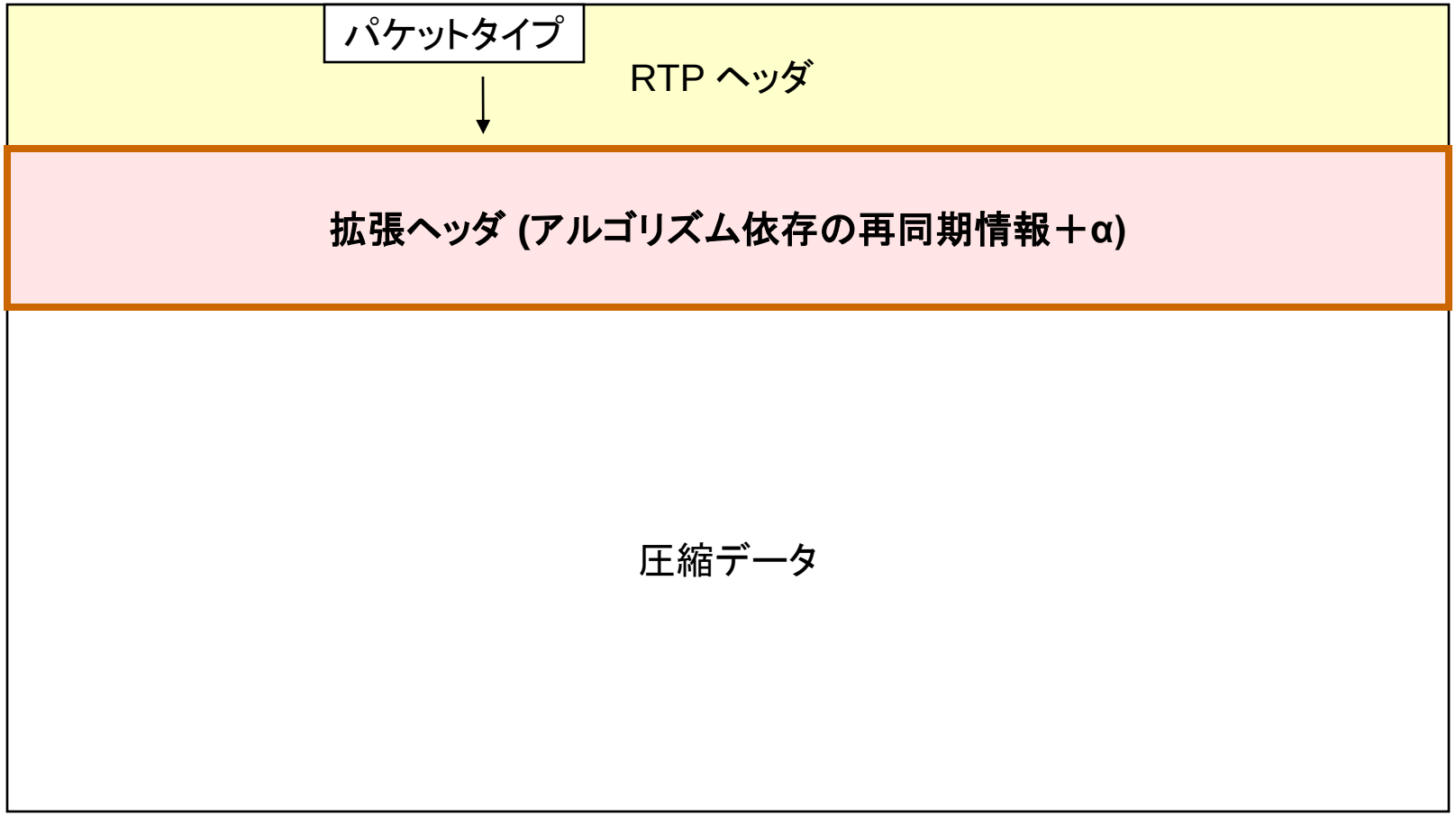
シーケンスナンバ: **パケット廃棄の検出**

タイムスタンプ: 同期再生 (メディア内同期)

Mビット: フレーム境界の通知

SSRC: ストリームの識別

RTPペイロードフォーマット

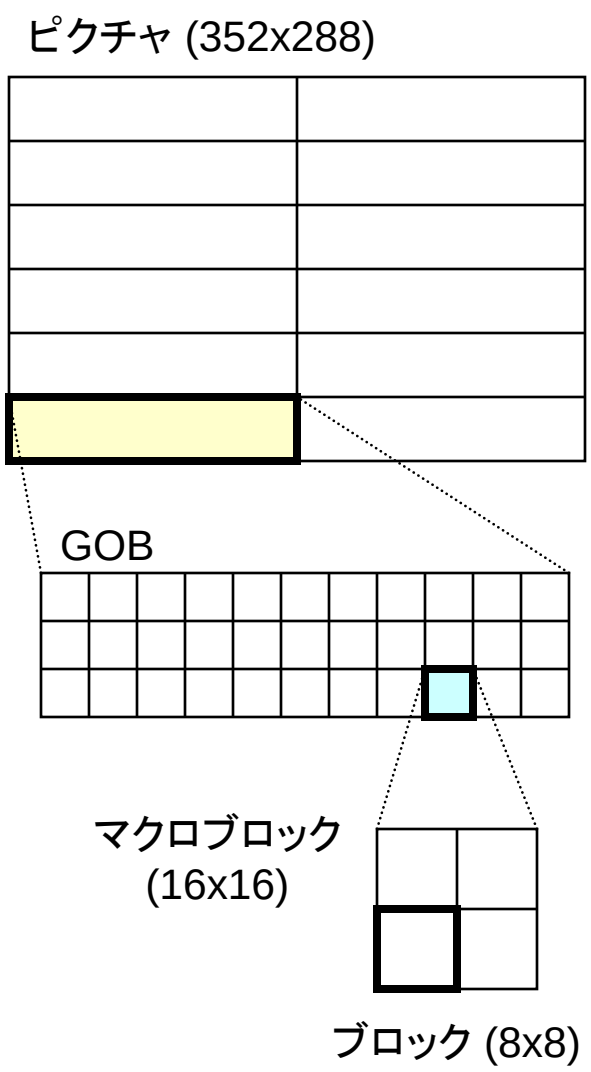


符号化アルゴリズム毎に、さまざまなペイロードフォーマットが決められている (RFC)

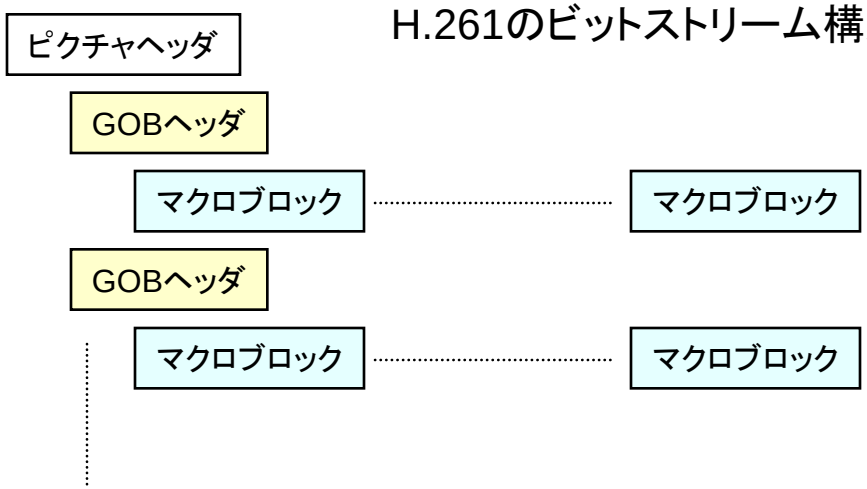
ペイロードフォーマットのRFC一覧

RFC 番号	符号化アルゴリズム
1890	各種音声符号化 (G721, G722, G728 等)
2032	ITU-T H.261
2190	ITU-T H.263
2250	ISO/IEC MPEG1/MPEG2 Video/Audio
2429	ITU-T H.263+
2435	ISO JPEG
3016	ISO/IEC MPEG4 Audio/Visual
3047	ITU-T G.722.1
3119	ISO/IEC MP3 Audio
I-D	ISO JPEG-2000

RFC2032 (H.261)



H.261のビットストリーム構造



再同期位置？

→ GOBヘッダ、あるいはマクロブロック

マクロブロックにまたがって継承される情報？

→ マクロブロックアドレス、動きベクトル、量子化ステップサイズ、等。

→ **これらを再同期情報としてコピー**

RFC2032 (H.261)

RTP ヘッダ:

フレームの最後で、M ビットを 1にセット。
タイムスタンプの解像度は 90kHz。

H.261ヘッダ (4バイト):

SBIT	EBIT	I	V	GOBN	MBAP	QUANT	HMVD	VMVD

SBIt, EBIT: 先頭、最終バイトの有効ビットの位置 (H.261ではバイトアラインが行われないため)。
I: イントラフレーム or インターフレーム。
V: 動きベクトルが使われている or 使われていない。
GOBN: パケットの先頭のマクロブロックのGOB番号。
MBAP: パケットの先頭のマクロブロックのマクロブロックアドレス。
QUANT: パケットの直前で有効だった量子化ステップサイズ。
HMVD, VMVD: パケットの先頭のマクロブロックの動きベクトル。

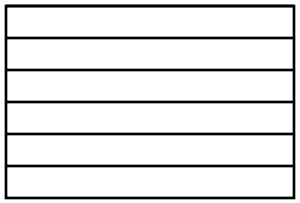
} 再同期情報

圧縮データのフラグメンテーション:

ピクチャ、GOB、あるいはマクロブロック境界にアライン

RFC2190 (H.263)

H.261の機能拡張 (半画素動き検出、GOBのライン化、ほか) →



H.263 特有の機能 (オプション):

ベクトル探索範囲の拡大 (Annex D):
算術符号化 (Annex E): ハフマン符号化の代替オプション。
アドバンス予測（Annex F): 8x8ブロック単位の動き補償、オーバーラップ動き補償。
PB フレーム (Annex G): B ピクチャの簡易版。

↓

H.263 用ペイロードフォーマット:

Mode A: ピクチャ、もしくはGOB境界にアライン。
Mode B: PB フレームなし、マクロブロック境界にアライン。
Mode C: PB フレームあり、マクロブロック境界にアライン。

↓

Mode A の利用が推奨。

RFC2190 (H.263)

H.263 ヘッダ Mode A (4バイト): GOB 単位

F	P	SBIT	EBIT	SRC	I	U	S	A	reserved	DBQ	TRB	TR

F: 0 の場合 mode A、1 の場合 mode B/C。
P: 0 の場合 I/P フレーム、1 の場合 PB フレーム。
SRC: ピクチャ解像度。
U: Annex D オプション (ベクトル探索範囲拡大) の on/off。
S: Annex E オプション (算術符号化) の on/off。
A: Annex F オプション (アドバンス予測) の on/off。
DBQ: PB フレームオプションの差分量子化パラメータ。
TRB、TR: PBフレームオプションのテンポラルリファレンス。

再同期情報

Mode B (8バイト): マクロブロック単位、PB オプションなし

GOB番号、量子化ステップサイズ、マクロブロックアドレス、動きベクトルの複製。
差分量子化パラメータ、テンポラルリファレンスの削除。

Mode C (12バイト): マクロブロック単位、PB オプションあり

Mode A & B に使用されるすべてのフィールドから構成。

RFC2429 (H.263+)

H.263の機能拡張

インターネット用途に有効な H.263+ の拡張機能:

スライス構造 (Annex K): GOB の代替。固定されたGOBとは異なり、スライス幅を動的に変更可能、スライススタートコードでバイトアラインされる。

独立セグメント復号 (Annex R): セグメント (GOB /スライス) 単位で独立して復号可能。動きベクトルの探索範囲はセグメント内に限定。

スケーラビリティ（Annex O): Temporal, SNR & spatial scalability。時間解像度と空間解像度の階層化、SNR エンハンスメント。

参照ピクチャ選択モード (Annex N): 参照ピクチャの動的切り替え。エラー通知によるリカバリ。

ペイロードフォーマットの工夫 (H.261/H.263 用とはかなり違う):

ヘッダの簡素化。

ピクチャヘッダの複製の挿入。

スケーラビリティは、個々の階層を独立したストリームとしてパケット化。

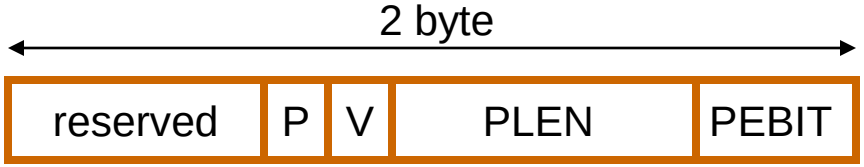
RFC2429 (H.263+)

RTP ヘッダ:

フレームの最後で、M ビットを 1にセット。

タイムスタンプの解像度は 90kHz。

H.263+ヘッダ



P: スタートコード (ピクチャ、GOB、スライス) から始まる場合、1にセット。

V: ビデオ冗長符号化が使われる場合、1にセット。

PLEN: ピクチャヘッダが挿入されている場合、その長さ (バイト単位)。

PEBIT: ピクチャヘッダの最後のバイトで無視されるビット数。

} 再同期情報

圧縮データのフラグメンテーション:

制約無し (Pビットで識別)。

P=0 で前パケットが廃棄された場合、受信パケット中のスタートコードをサーチし、それを再同期ポイントとする。

RFC3016 (MPEG-4 Video/Audio)

MPEG-4 Video と H.263+ の対比:

再同期マーカ: 17ビットの再同期マーカを先頭に、マクロブロック群の固まりを構成 (ビデオパケット)。

→ H.263+ のスライス構造。

ピクチャヘッダのコピー: フラグに応じて、ビデオパケット単位にピクチャヘッダ (VOP ヘッダ) を複製。

→ H.263+ ペイロードフォーマットのピクチャヘッダ複製機能。

データパーティショニング: マクロブロック情報を動きベクトルとテクスチャ情報に分け、モーションマーカ (17ビット) を挿入して分離。

→ H.263++ で採用。

リバーシブルVLC: DCT係数のハフマン符号で、両方向から復号可能な VLC 。

→ インタネットではあまり大きな意味を持たない。

スケーラビリティ: H.263+ と同様。

形状符号化: JBIG 拡張としてのオブジェクト形状の符号化。

→ MPEG-4 独自。廃棄対策は、再同期情報の挿入。

RFC3016 (MPEG-4 Video/Audio)

RTP ヘッダ:

フレーム (VOP) の最後で、M ビットを 1にセットする。

MPEG-4 Video 用ヘッダ

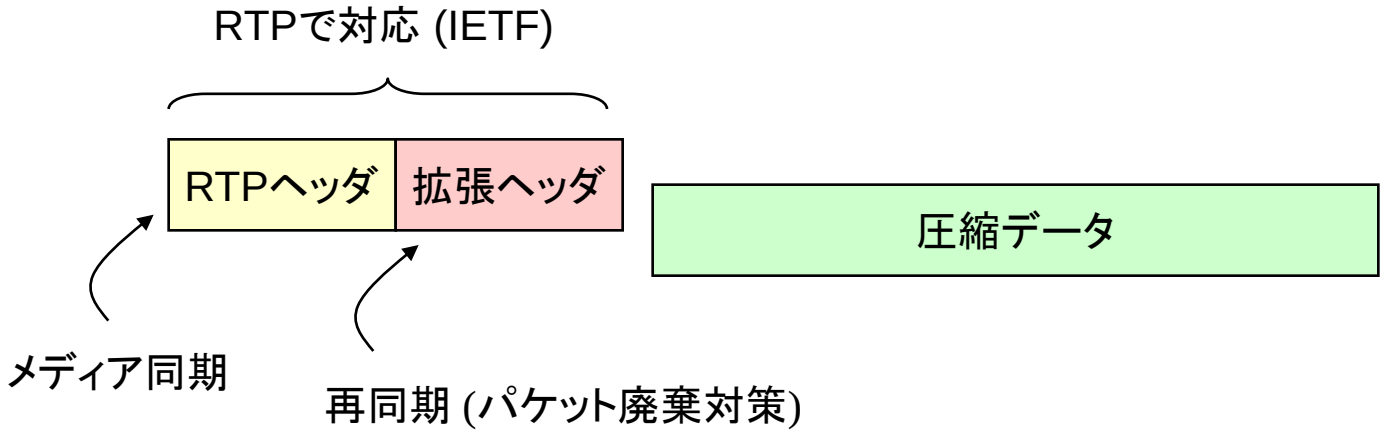
なし。

圧縮データのフラグメンテーション:

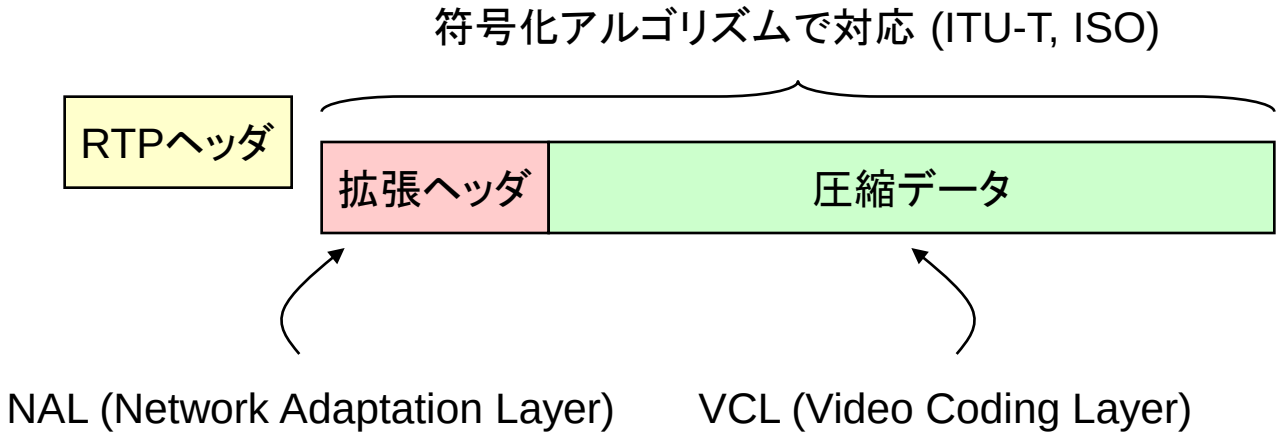
構成情報と GOV はペイロードの先頭に来なければならない

ペイロードフォーマットの歴史

昔：H.261、H.263



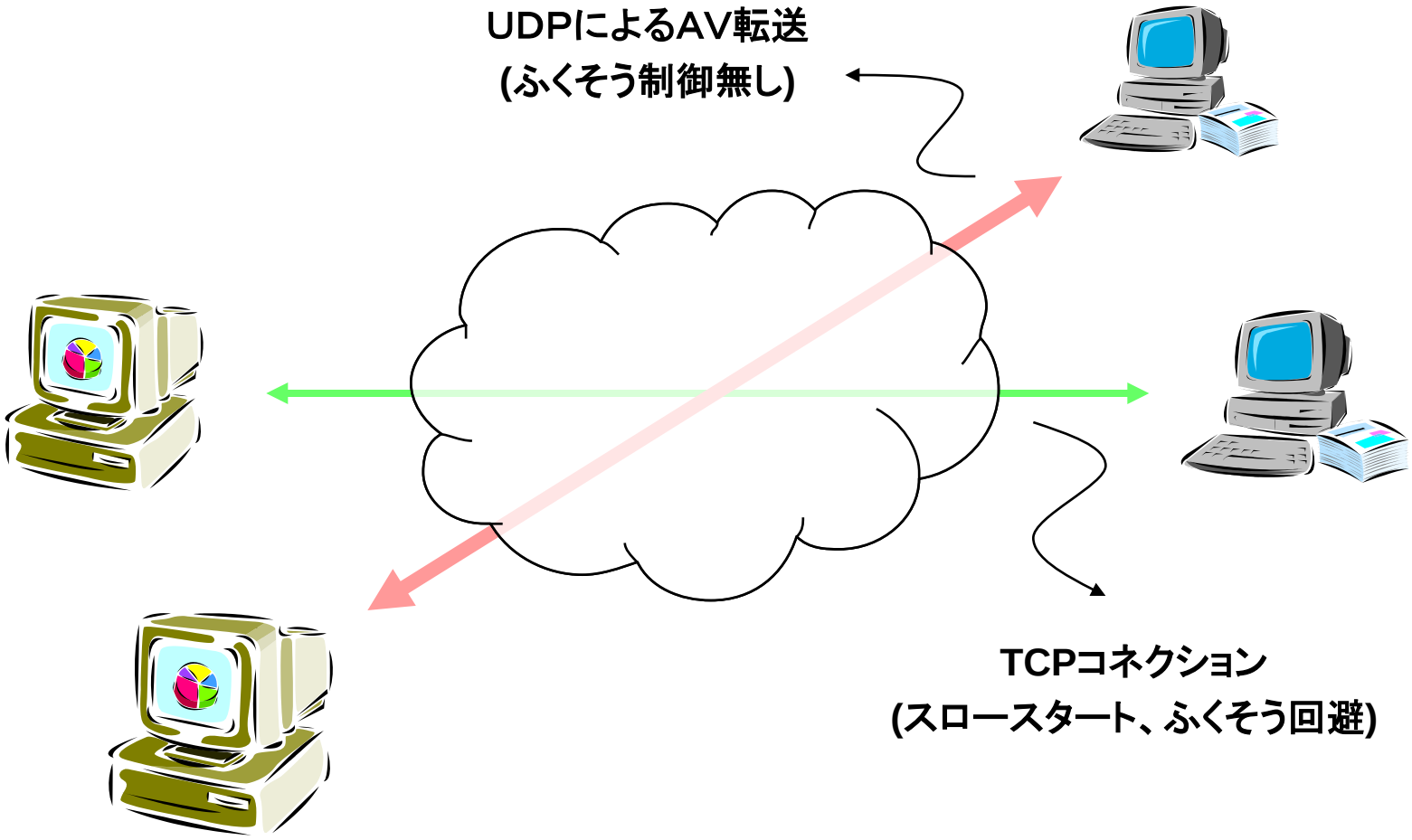
最近：H.263+、MPEG-4、H.264



TCPフレンドリ
(フロー制御)

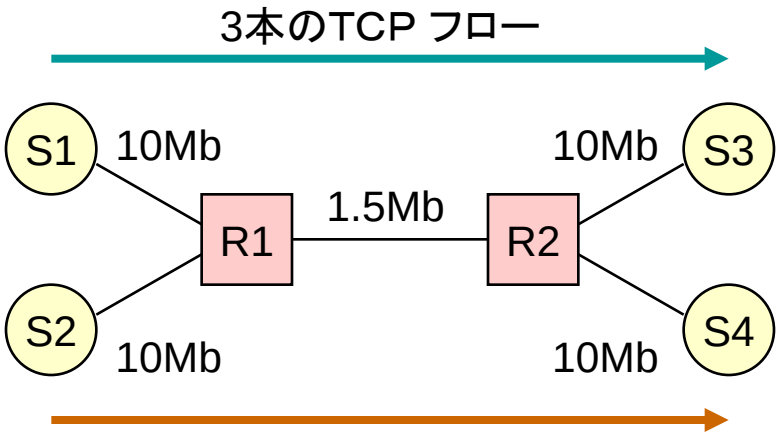
フロー制御の必要性 (1)

- UDP 帯域が増えると TCP 帯域はどうなるか？

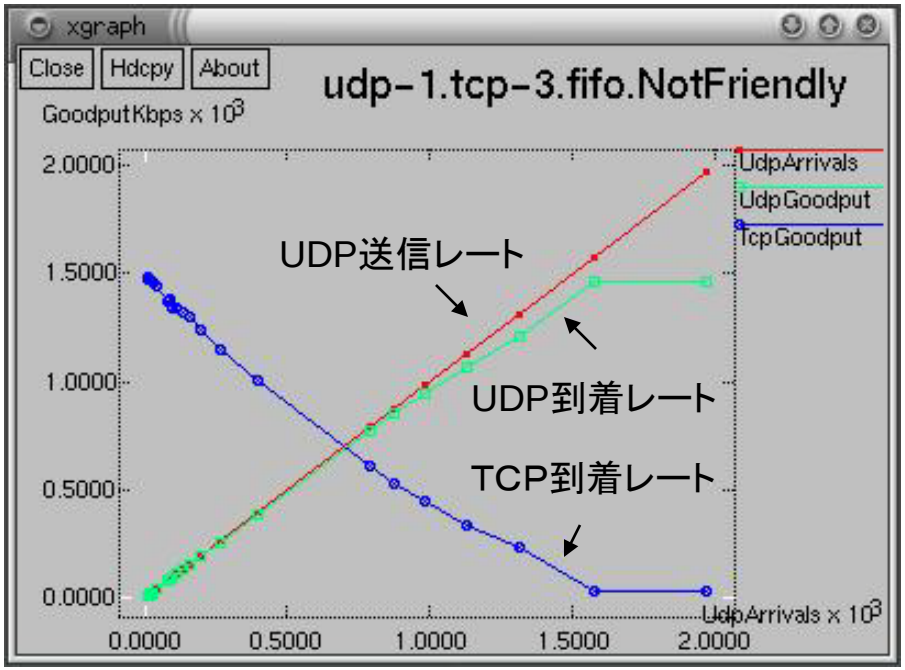


フロー制御の必要性 (2)

- UDP 帯域が増えると TCP 帯域はどうなるか？



UDP の送信レートの増加に伴い、
TCP のスループットが低下する。
リンク容量を越えた UDP は廃棄。



ネットワークシミュレータ

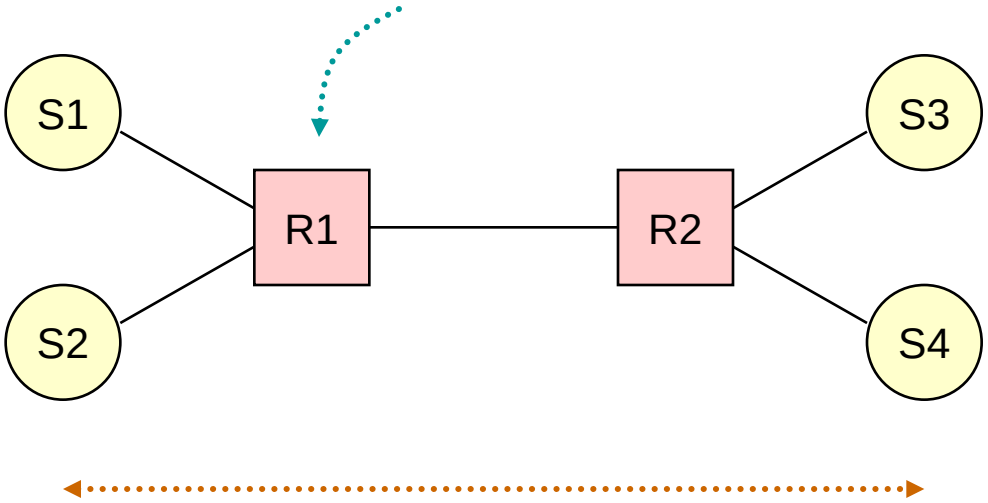
S.Floyd: “Promoting the Use of End-to-End Congestion Control,” IEEE/ACM Trans on Networking, 1999.

フロー制御の必要性 (3)

- どこで解決するか？

⇒ diffserv、MPLS

① フローの差別化 (at ルータ: ネットワーク層)



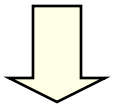
② End-to-End 制御 (at 端末: トランスポート・アプリケーション層)

⇒ TCPフレンドリ

TCPフレンドリ

- **TCP と UDP をどのように共存させるか？**

UDP レートを TCP レートと等しくなるように符号量制御する。



方法1: UDP に対して TCP と同じふくそう制御メカニズムを適用する。

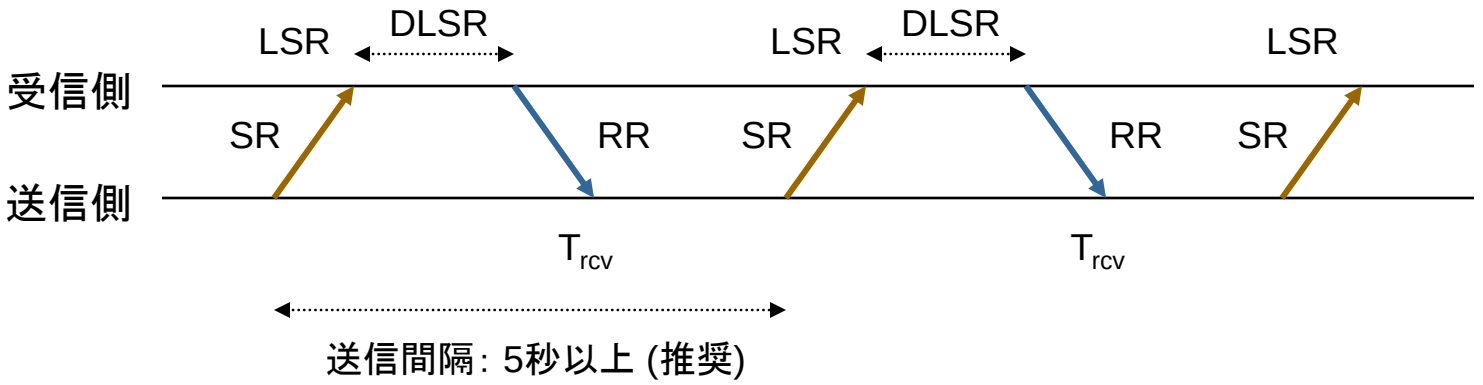
問題：レート変動が激しすぎて、AV アプリケーションには適用できない。

方法2: 測定可能なパラメータから TCP と等価なレートを見積もり、そのレートに適合するように UDP フローを制御する。

問題：TCP と等価なレートをどのように推定するか？

RTCP-RR: Report Block

- 受信側から送信側に返される統計情報量



廃棄率: p = **廃棄パケット数** / 送信パケット数

ラウンドトリップ遅延: $RTT/2 = T_{rcv}$ - **DLSR** - **LSR**

パケットサイズ: 送信側で測定可能

→ フロー制御 (TCPフレンドリ)

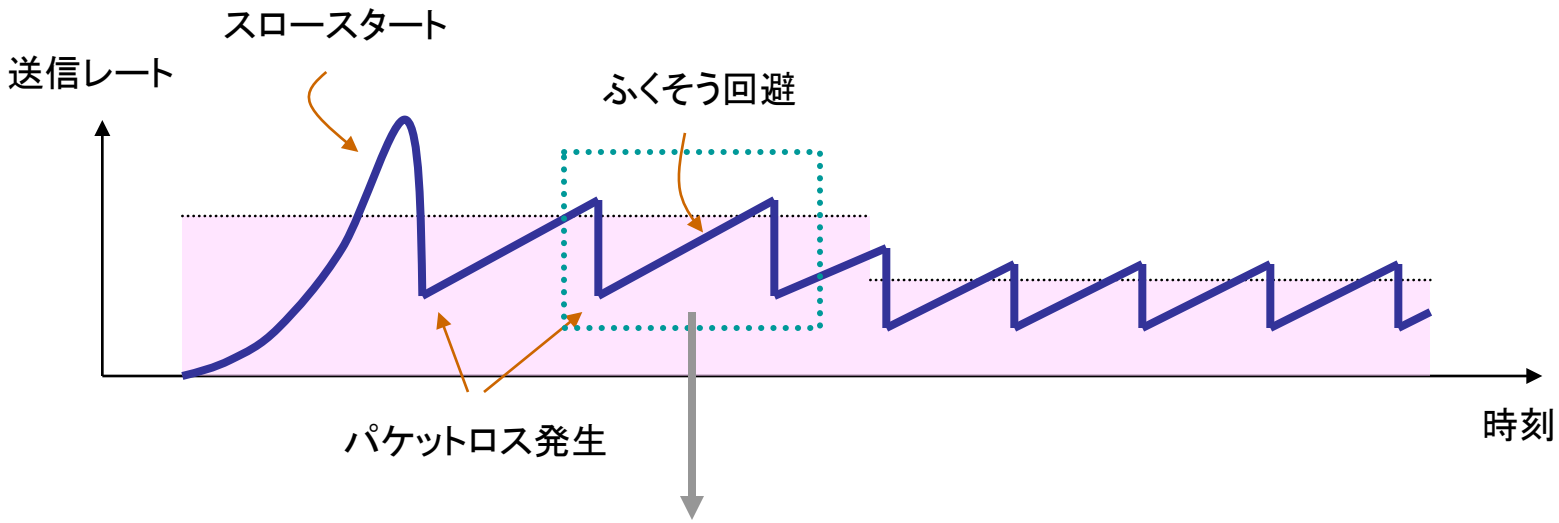
(注) RTCP 自体は、具体的なふくそう制御アルゴリズムは何も決めていない (標準というのはそういうもの)

RFC 1889

TCPのモデル

- TCP のモデル化

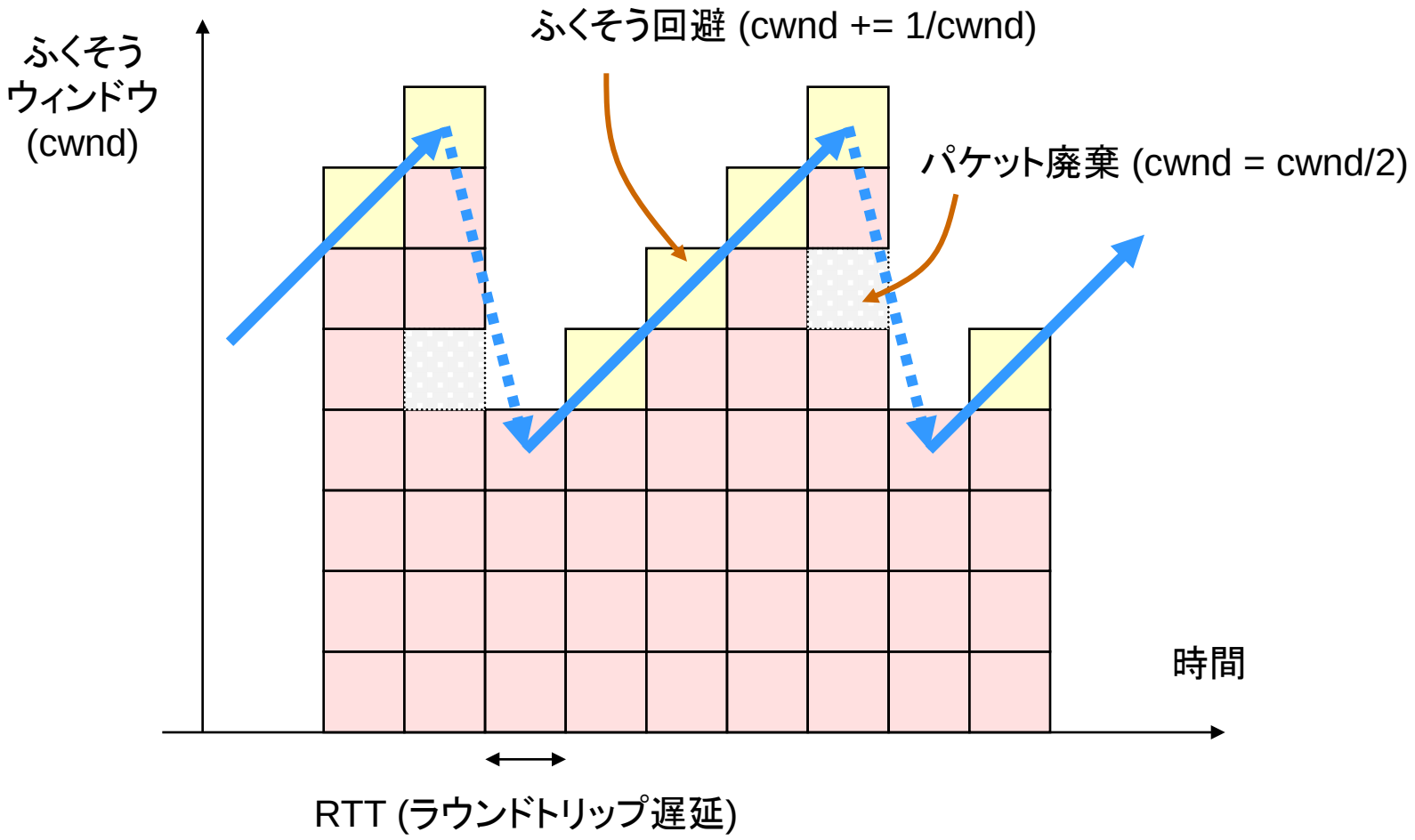
TCP Reno の場合



TCPの定常状態における
ふるまいをモデル化

モデル (1)

- TCP Reno のふくそう回避アルゴリズムのモデル化 (1)



S.Floyd: “Promoting the Use of End-to-End Congestion Control,” IEEE/ACM Trans on Networking, 1999.

モデル (1)

定常状態:

1個のパケット廃棄

RTT ラウンド	1	2	3	.....	W/2	W/2+1	W/2+2
ふくそうウィンドウ	W/2	W/2+1	W/2+2	.....	W	W/2	W/2+1

解析:

(1) パケット廃棄の発生間隔 $\frac{W}{2} \cdot RTT$

(2) その期間の送信パケット数 $\frac{W}{2} + \left(\frac{W}{2} + 1\right) + \cdots + W = \frac{3}{8}W^2$

(3) パケット廃棄率 p $p = \frac{8}{3W^2}$ (総送信パケットのひとつが廃棄)

モデル (1)

TCP フレンドリなパケット送信レート:

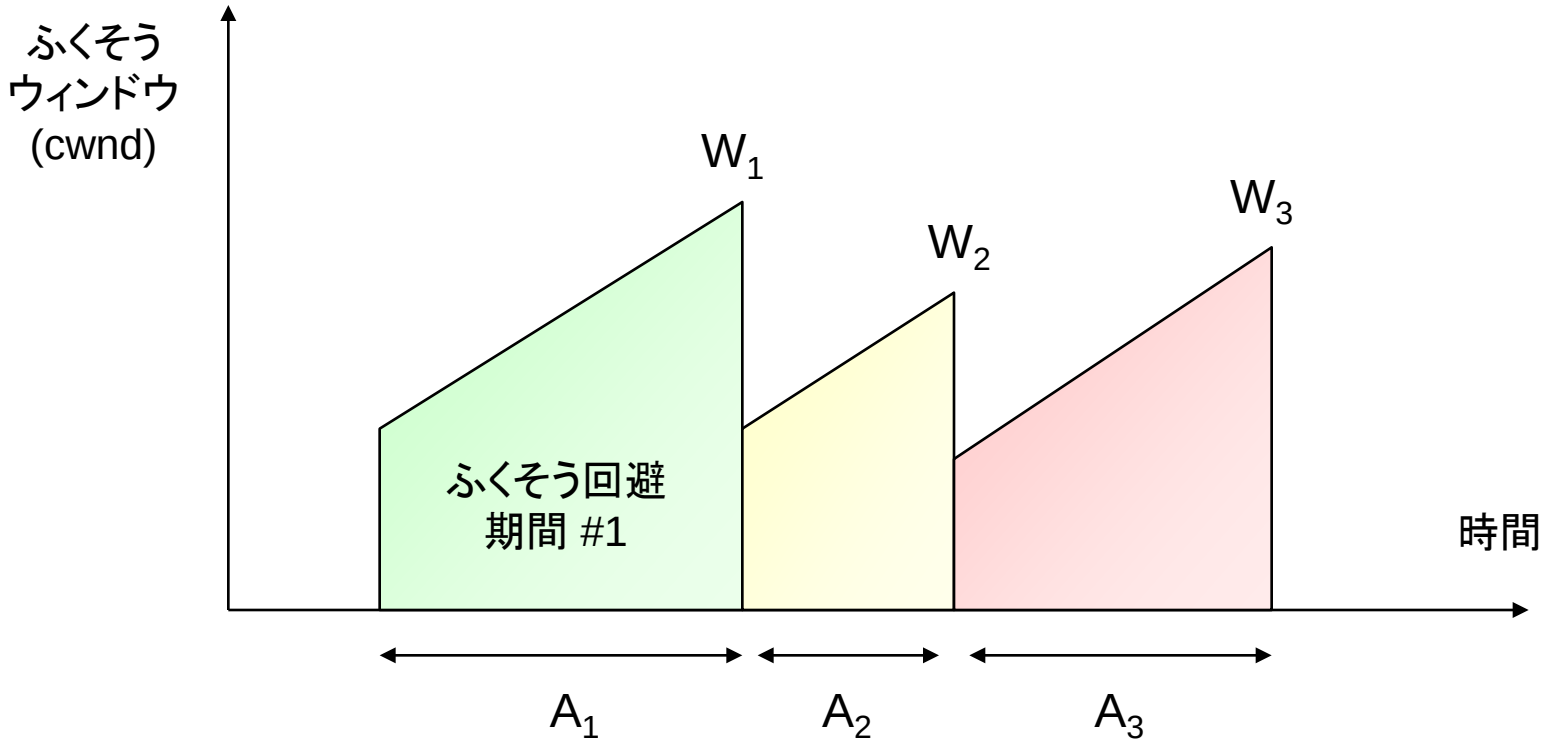
(1回のふくそう回避期間に送信されるパケット数の期待値)

$$\therefore R = \frac{\frac{3}{8}W^2}{\frac{W}{2} \cdot RTT} = \frac{1}{RTT}\sqrt{\frac{3}{2p}}$$

ラウンドトリップ遅延　　パケット廃棄率

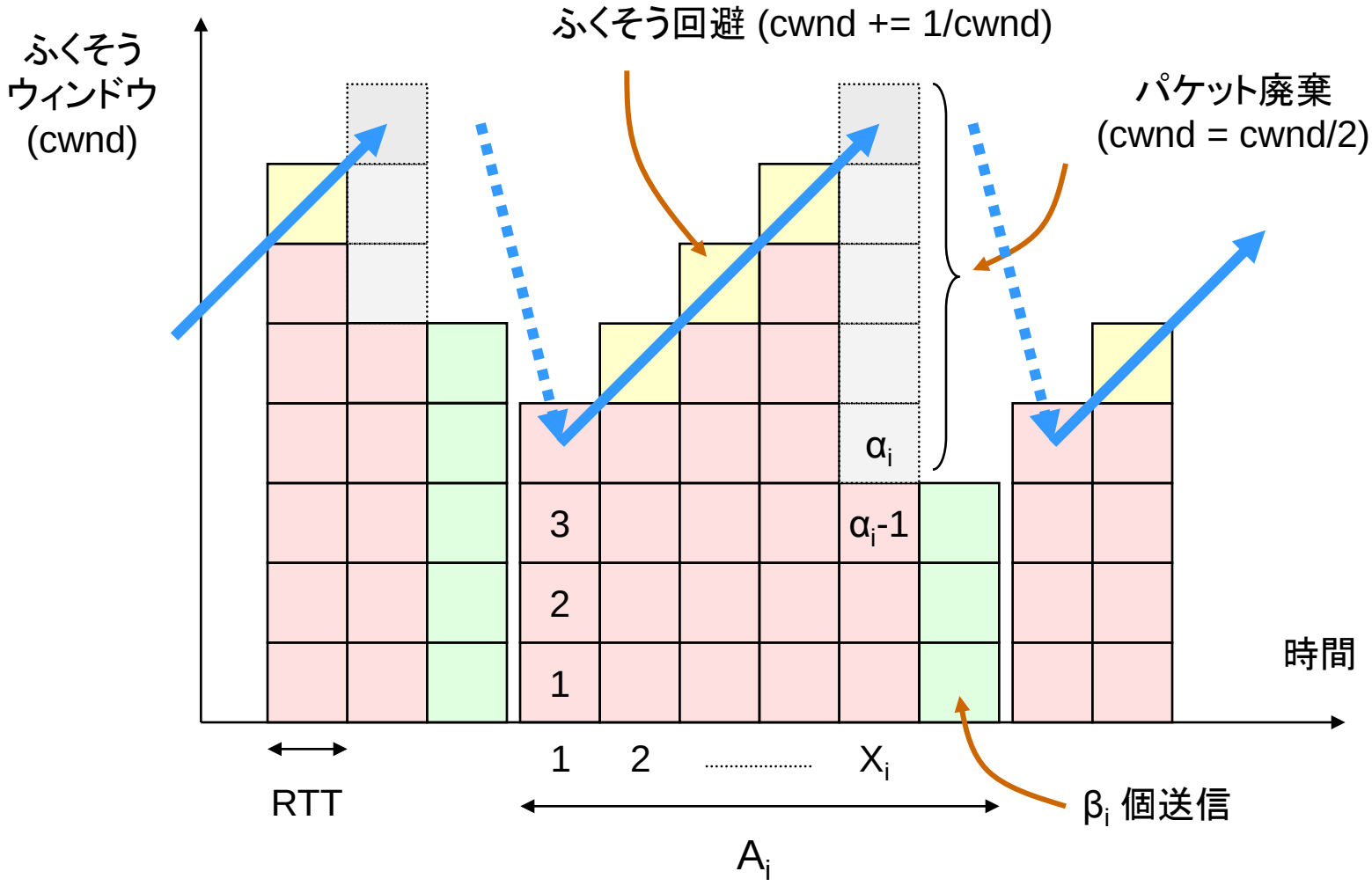
モデル (2)

- TCP Reno のふくそう回避アルゴリズムのモデル化 (2)



モデル (2)

- TCP Reno のふくそう回避アルゴリズムのモデル化 (2)



J.Padhye et al: “Modeling TCP Throughput: A Simple Model and its Empirical Validation,” ACM SIGCOMM98.

モデル (2)

パラメータの定義:

i	i 番目のふくそう回避期間
A_i	経過時間
Y_i	総送信パケット数 (廃棄パケットを含む)
X_i	廃棄発生までの RTT ラウンド数
W_i	廃棄発生時の RTT ラウンドのふくそうウィンドウの値
α_i	廃棄発生時までの送信パケット数 + 1
β_i	廃棄発生直後の RTT ラウンドの送信パケット数
p	パケット廃棄率
RTT	ラウンドトリップ遅延

仮定: 同一 RTT ラウンドの廃棄発生後のパケットはすべて廃棄

モデル (2)

解析:

(1) Y_i、W_i、α_i の関係　　期待値

$$Y_i = \alpha_i + W_i - 1 \longrightarrow E[Y] = E[\alpha] + E[W] - 1$$

(2) E[α] (廃棄発生まで連続して送信可能なパケット数の期待値) の算出

$$E[\alpha] = \sum_{k=1}^{\infty} k \cdot P[\alpha = k] = \sum_{k=1}^{\infty} k \cdot (1-p)^{k-1} p = \frac{1}{p}$$

(3) W_i と X_i の関係 (ふくそうウィンドウの増加)

$$W_i = \frac{W_{i-1}}{2} + X_i \longrightarrow E[X] = \frac{1}{2} E[W]$$

(4) Y_i、W_i、X_i、β_i の関係 (ふくそうウィンドウの積分)

$$Y_i = \sum_{k=0}^{X_i - 1} \left(\frac{W_{i-1}}{2} + k \right) + \beta_i \longrightarrow E[Y] = \frac{E[X]}{2} \left(\frac{3}{2} E[W] - 1 \right) + E[\beta]$$

モデル (2)

解析:

(5) E[β] (最終ラウンドの送信パケット数の期待値) の算出

$$E[\beta] = \frac{1}{2} E[W]$$

(6) E[W] とパケット廃棄率 p の関係 ((1) ～ (5) 式より)

$$E[W] = 1 + \sqrt{1 + \frac{8(1-p)}{3p}} \approx \sqrt{\frac{8}{3p}} \qquad (p \to 0)$$

⇨ モデル(1) に一致

(7) E[A] (ふくそう回避時間の期待値) の算出

$$E[A] = RTT \cdot (E[X] + 1) = \cdots = RTT \cdot \left(\frac{3}{2} + \sqrt{\frac{2(1-p)}{3p} + \frac{1}{4}} \right)$$

モデル (2)

TCP フレンドリなパケット送信レート:

(1回のふくそう回避期間に送信されるパケット数の期待値)

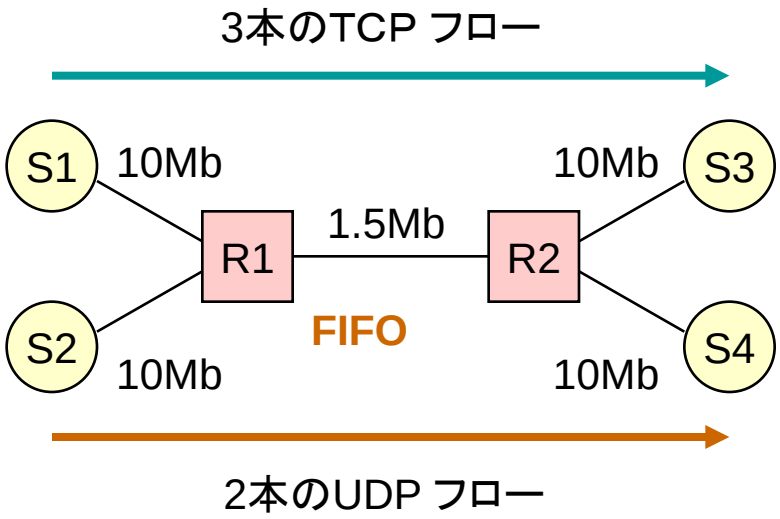
$$R = \frac{E[Y]}{E[A]} = \frac{\frac{1}{p} + \sqrt{1 + \frac{8(1-p)}{3p}}}{RTT \cdot \left(\frac{3}{2} + \sqrt{\frac{2(1-p)}{3p} + \frac{1}{4}} \right)} \approx \frac{1}{RTT} \sqrt{\frac{3}{2p}}$$

モデル(1)に漸近 ($p \to 0$)

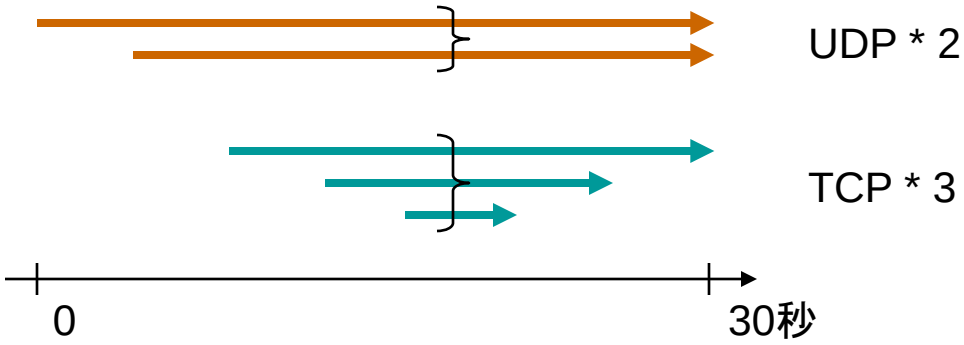
シミュレーション (1)

- TCP フレンドリの効果: シミュレーション条件

トポロジー



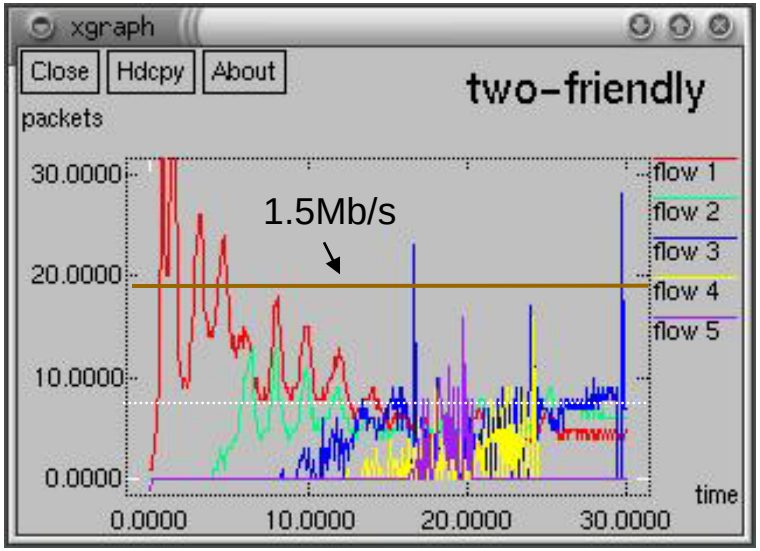
スケジュール



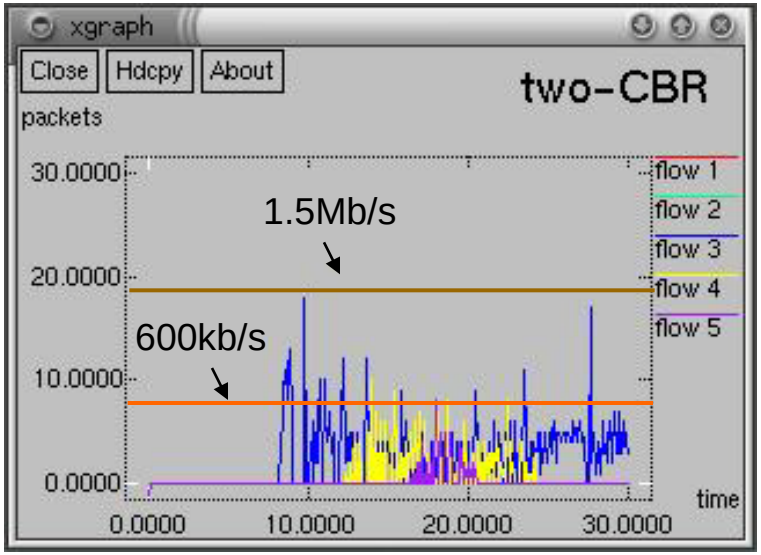
シミュレーション (2)

- TCP フレンドリの効果: シミュレーション結果

TCPフレンドリ対応



TCPフレンドリ未対応 (600kb/s * 2)



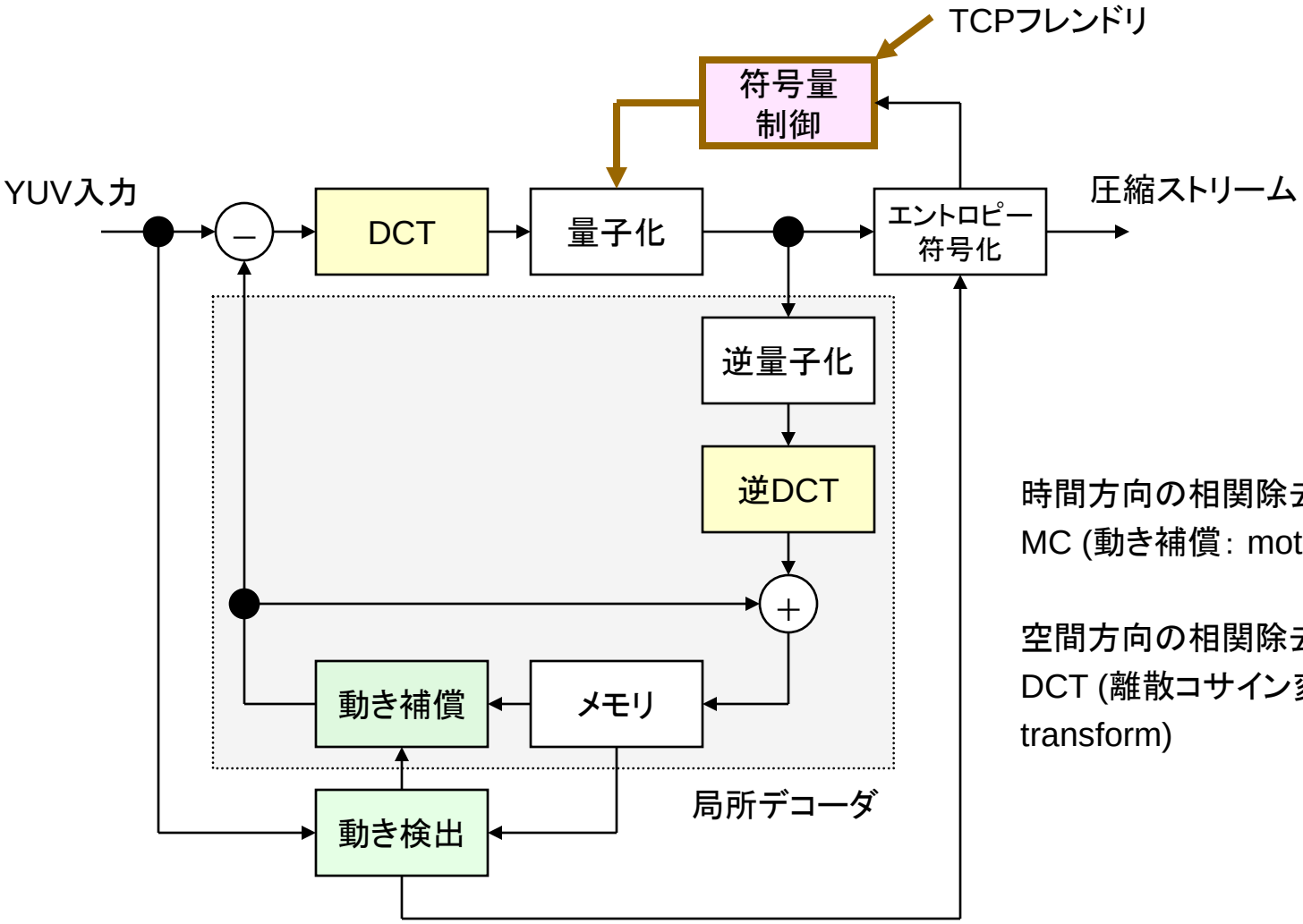
(0.1秒毎の送信パケット数のカウント)

	UDP#1	UDP#2	TCP#1	TCP#2	TCP#3
TCPフレンドリ	698kb/s	459kb/s	352kb/s	159kb/s	292kb/s
未対応	600kb/s	600kb/s	234kb/s	111kb/s	100kb/s

符号量制御
(メディア符号化側)

符号量制御 (1)

- 目標レートに合わせた量子化ステップサイズの制御



時間方向の相関除去:
MC (動き補償: motion compensation)

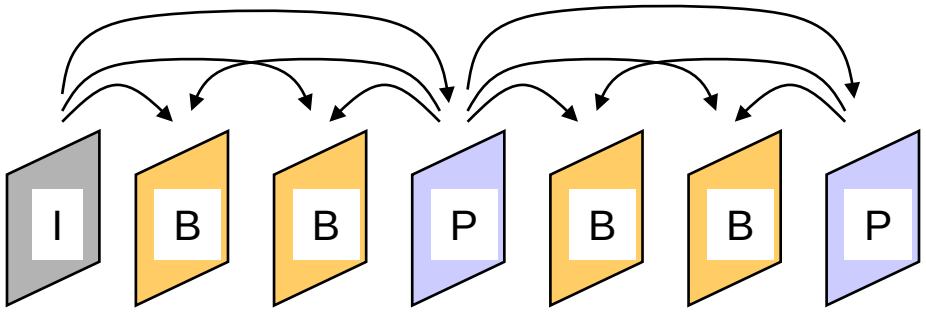
空間方向の相関除去:
DCT (離散コサイン変換: discrete cosine transform)

符号量制御 (2)

- 目標レートと量子化ステップサイズの関係

$$\log R = a \cdot \log Q + b \quad \Longleftrightarrow \quad R \cdot Q = const = X_{\{I,P,B\}}$$

ピクチャタイプ (I, P, B) 毎にほぼ一定

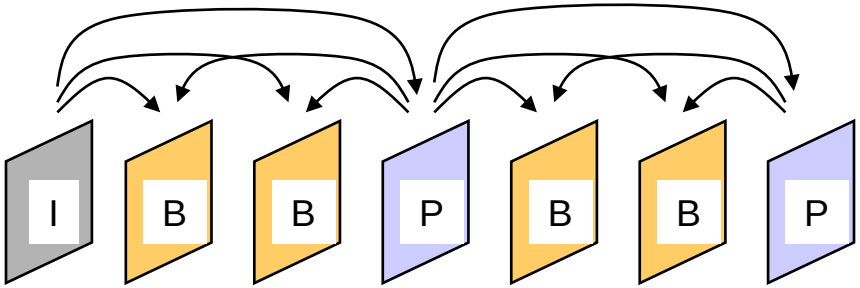


$X_{\{I,P,B\}}$ を事前に測定

目標レート R 確定 → 量子化ステップサイズ Q 確定

符号量制御 (3)

- TM5 アルゴリズム : ピクチャタイプに応じたレート配分



$$R_I = \frac{R_{remain}}{N_I + \left(\frac{X_P}{X_I}\right)\cdot N_P + \left(\frac{X_B}{X_I}\right)\cdot N_B}$$

$$R_P = \frac{R_{remain}}{N_P + \left(\frac{X_B}{X_P}\right)\cdot N_B}$$

$$R_B = \frac{R_{remain}}{N_B + \left(\frac{X_P}{X_B}\right)\cdot N_P}$$

ピクチャ毎の符号量配分を決めるアルゴリズム

R_{remain}: 残余ビットレート (初期値 : 目標レート R)
N_I, N_P, N_B: 各ピクチャの残余枚数
X_I, X_P, X_B: 一般的に $X_I > X_P > X_B$
1枚符号化する毎にパラメータを更新

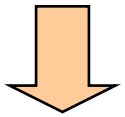
⇩

$R_I > R_P > R_B$ となる符号量配分 (準最適性の証明)

符号量制御 (4)

R-D Optimized Mode Selection

minimize $J = D + \lambda \cdot R$

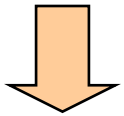


p: パケット廃棄率

p_C: パケット廃棄が伝播していない確率

パケットロス率を考慮したひずみの定義:

$$D = \underbrace{(1-p)\,p_C D_{no-loss}}_{\text{パケット廃棄が発生しない場合のひずみ}} + \underbrace{pD_{concealment}}_{\text{パケット廃棄によるひずみ}} + \underbrace{(1-p)(1-p_C)D_{propagate}}_{\text{パケット廃棄の影響の伝播によるひずみ}}$$



マクロブロック単位のイントラ・インターモード選択 (イントラ・アップデート)

“Rate-Distortion Optimization for JVT/H.26L Video Coding in Packet Loss Environment”, PV2002.

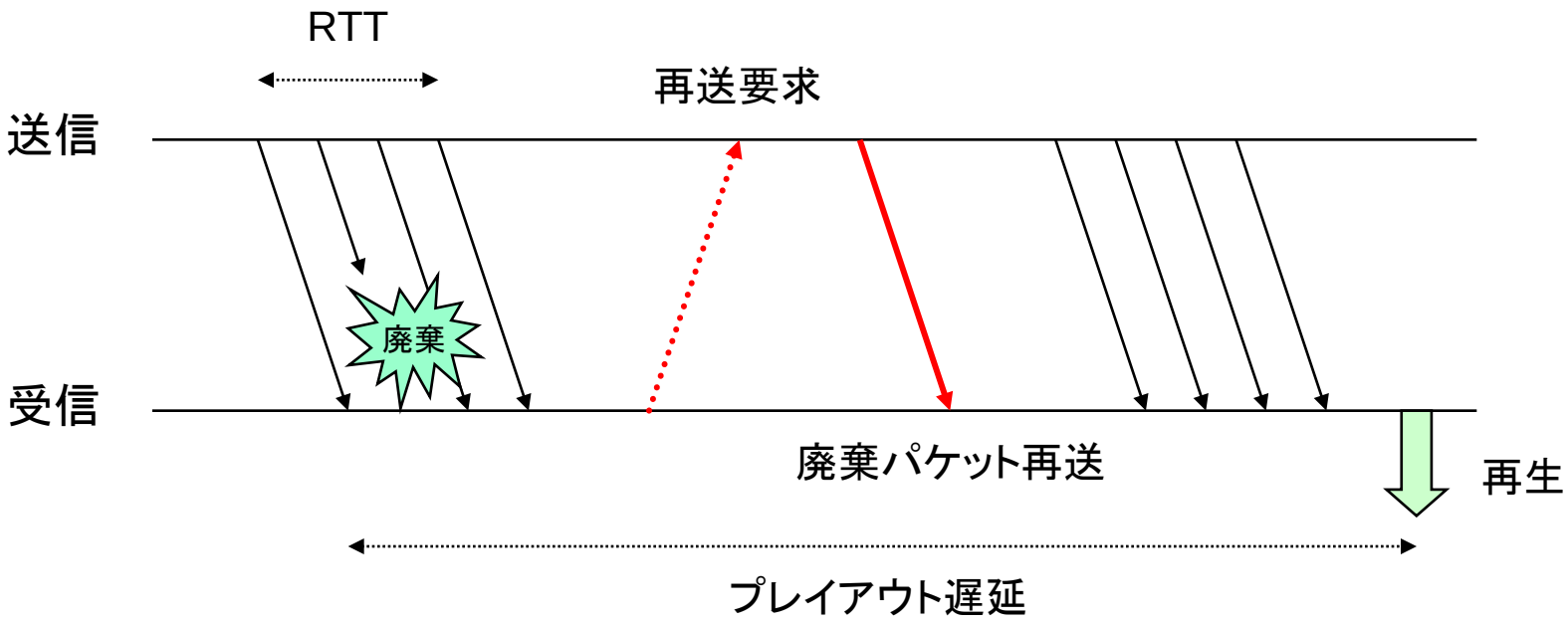
実践編

パケット再送

- **条件付きパケット再送**

  対象：遅延の小さい (RTTの小さい) 少人数会議・放送

  条件：RTT << プレイアウト遅延

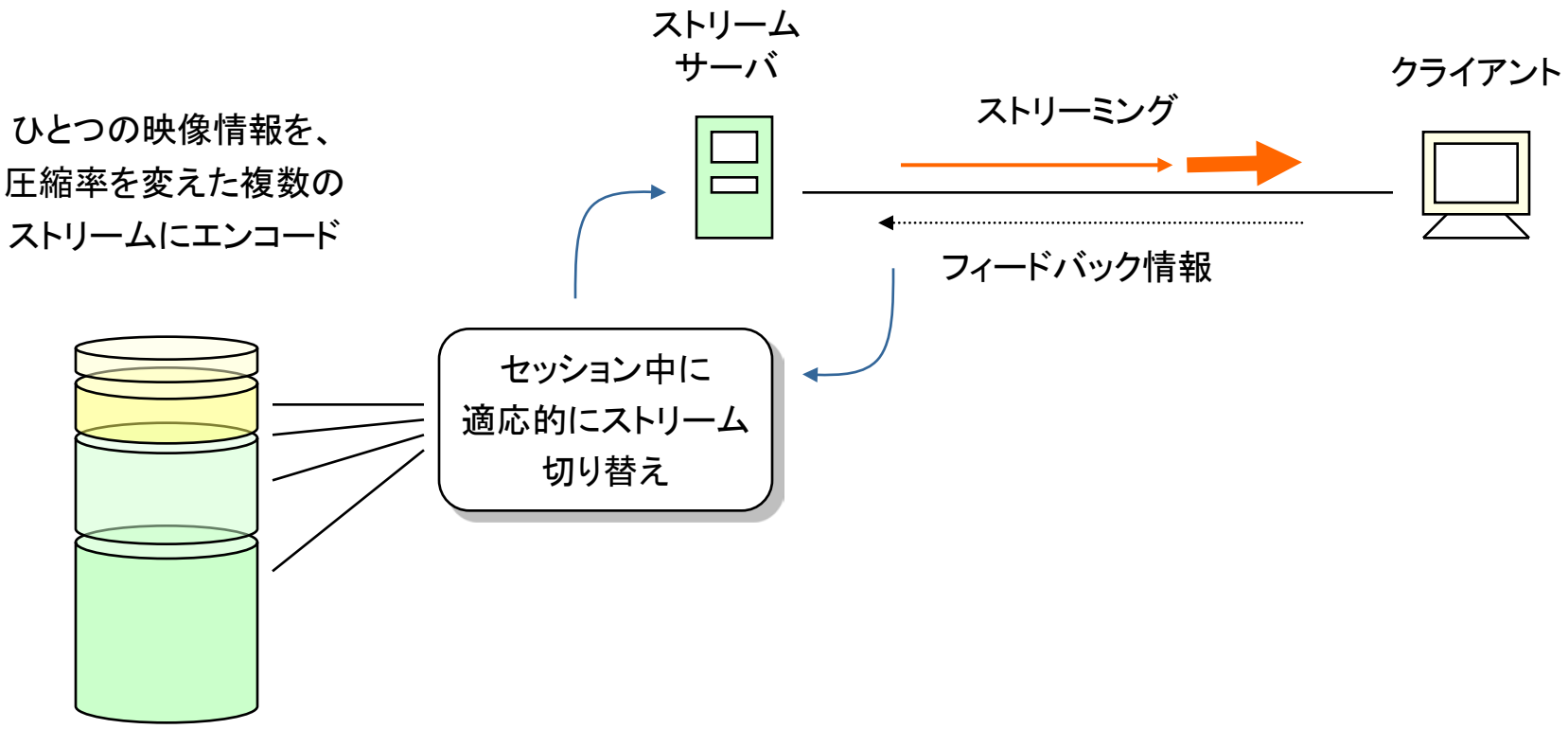


TCPフレンドリの応用 (1)

- **転送レートの切り替え**

対象：帯域幅に余裕のない片方向インターネット放送

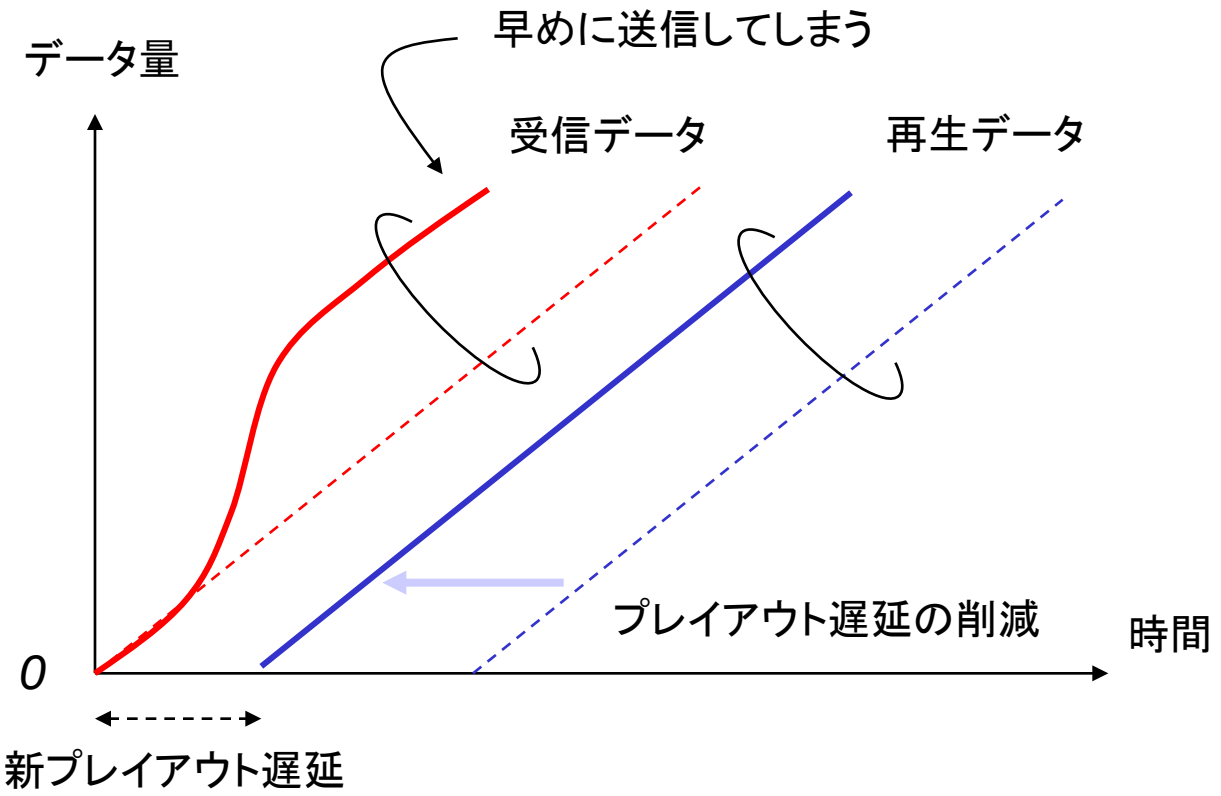
条件：転送レート > TCPフレンドリ見積もりレート



TCPフレンドリの応用 (2)

- プリフェッチング

対象：帯域幅に余裕のある片方向インターネット放送
条件：転送レート << TCPフレンドリ見積もりレート



前回資料参照.

アダプテーションのまとめ

まとめ

アダプテーション	RTP	RTCP	その他
メディア同期	RTP タイムスタンプ ⇒ メディア内同期	NTP タイムスタンプ＋ RTP タイムスタンプ ⇒ メディア間同期	NTP ⇒ システム間同期
パケット廃棄対策	RTP ペイロードフォーマット ⇒ 再同期		FEC パケット ディジタルファウンテン ⇒ 誤り訂正
フロー制御		パケット廃棄率 ラウンドトリップ遅延 ⇒ TCP フレンドリ	(RTP・RTCP 拡張)