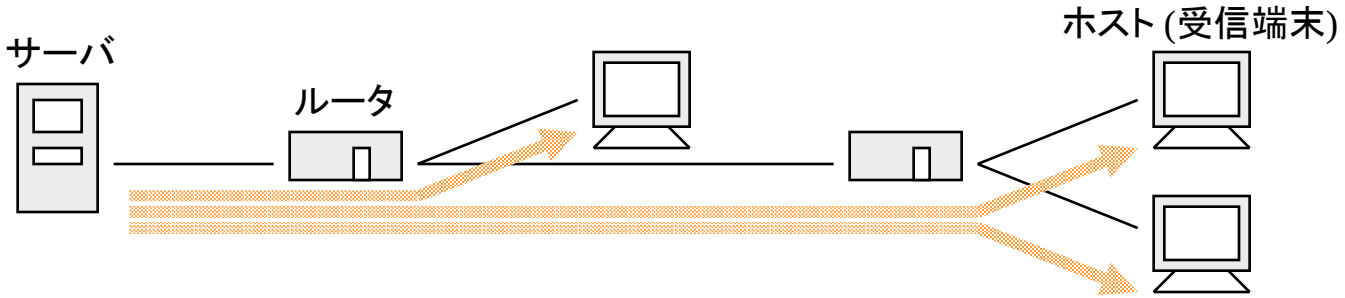
画像情報特論 (9)

- CDN/P2P、マルチキャスト

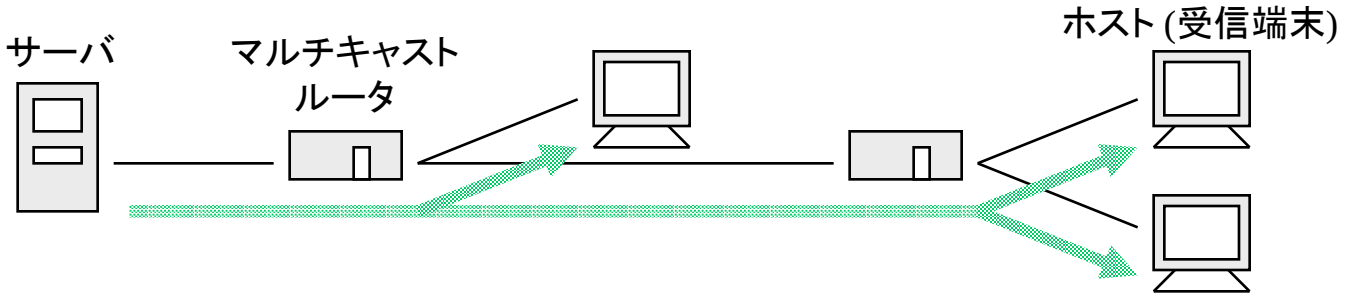
情報理工学専攻　甲藤二郎
E-Mail: katto@waseda.jp

IPマルチキャスト

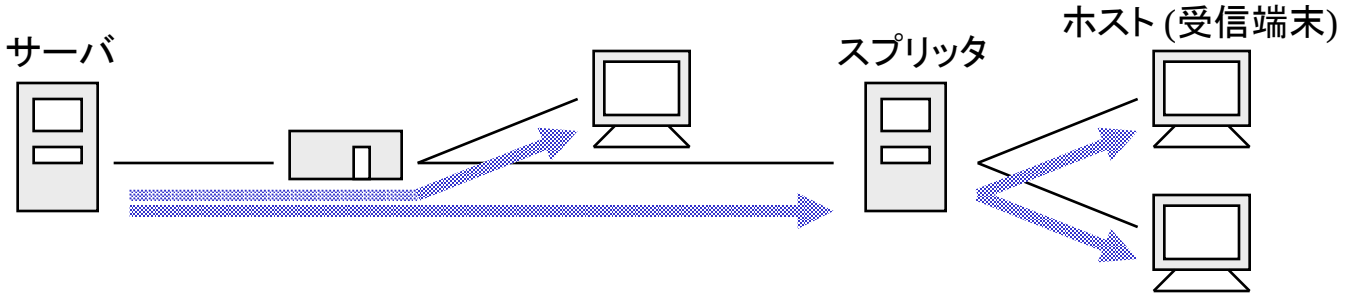
マルチキャスト



(a) ユニキャスト

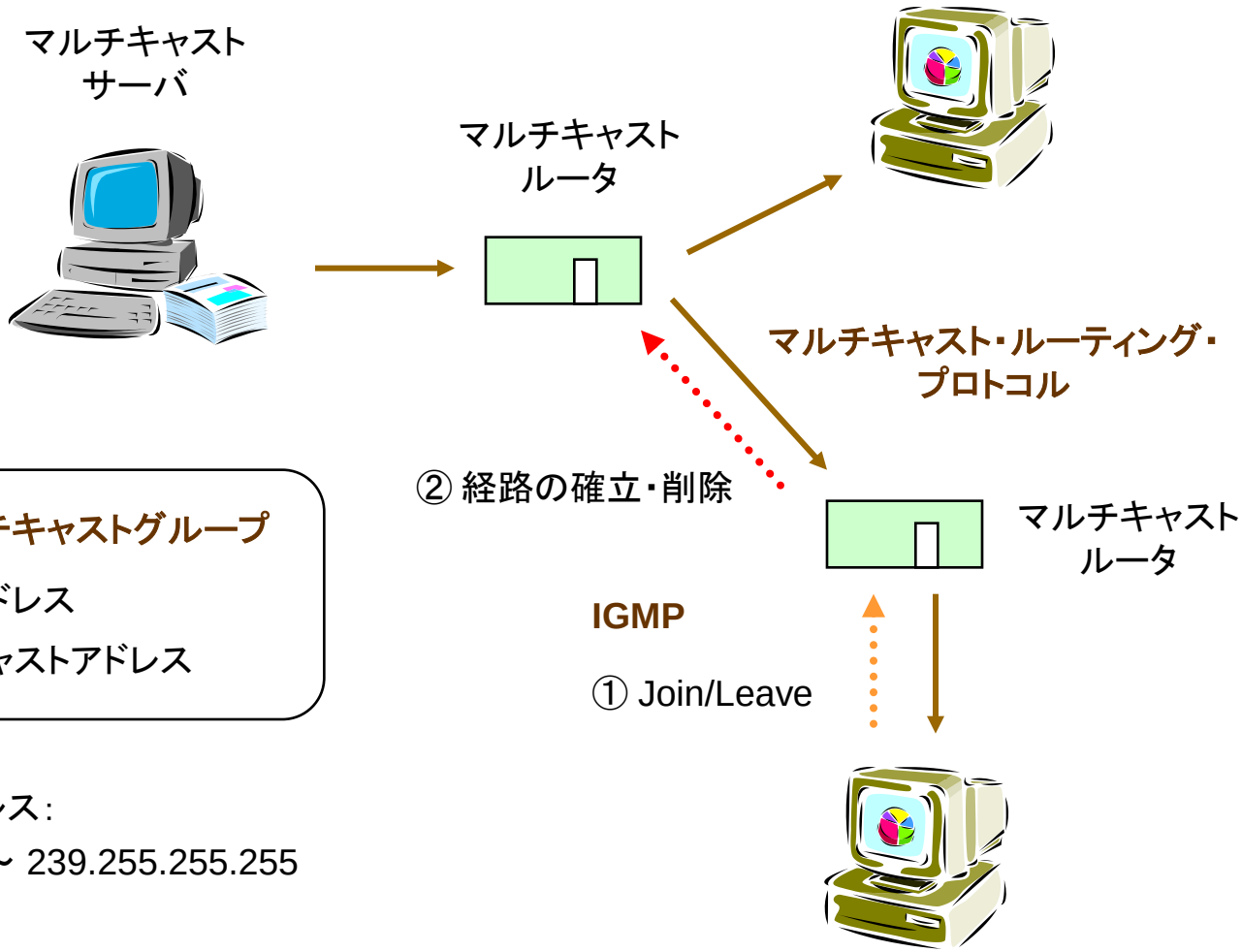


(b) マルチキャスト



(c) スプリッタ　(アプリケーション層マルチキャスト)

IPマルチキャスト (1)



(S,G): マルチキャストグループ

S: 送信者アドレス

G: マルチキャストアドレス

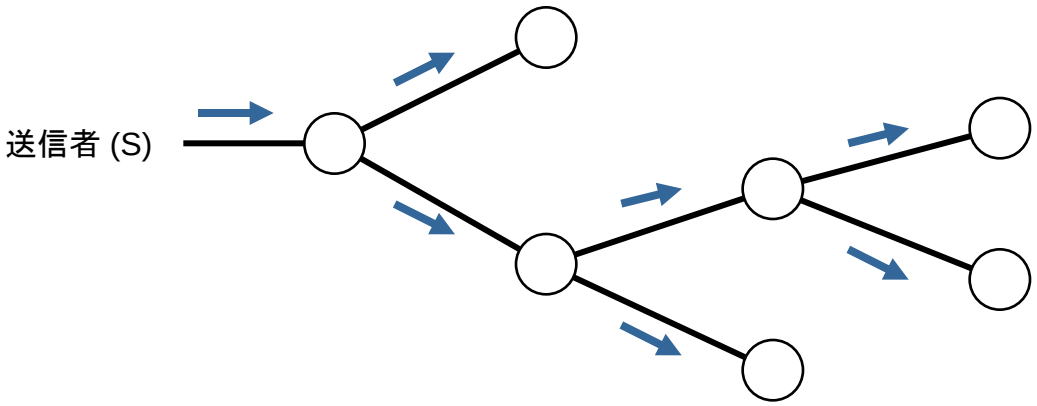
クラスDアドレス:

224.0.0.0 ～ 239.255.255.255

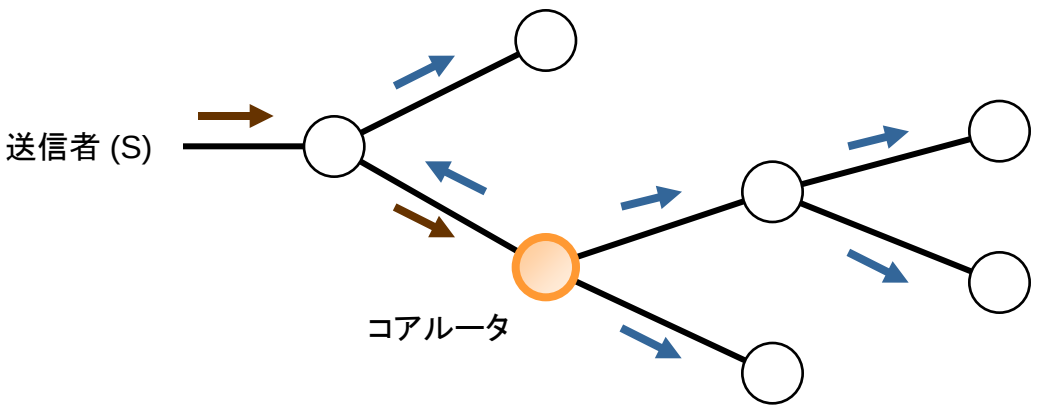
IPマルチキャスト (2)

- **Shortest Path Tree と Shared Tree**

Shortest Path Tree : (S, G)



Shared Tree : (*, G)



フラッディング:
各ルータは、パケットを受信したインタフェース以外のすべてのインタフェースにパケット転送。(S,G) エントリによる経路管理。

下流のルータは、状況に応じて転送停止・再開要求を出し、経路を確定。

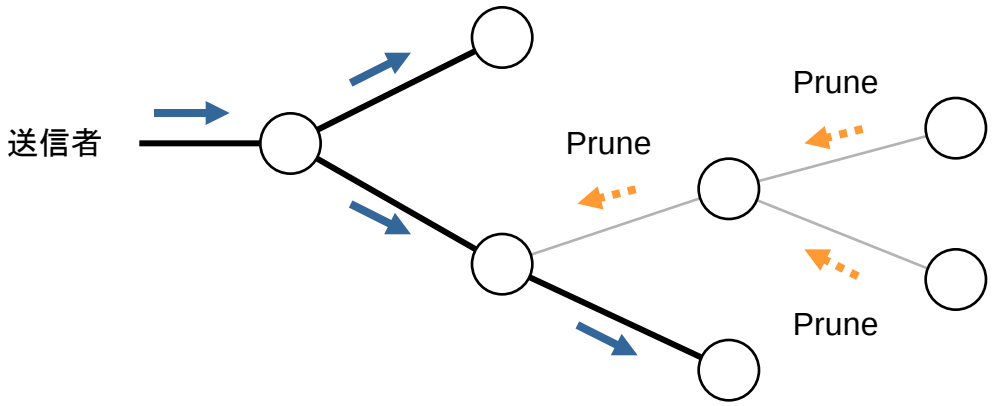
コアルータ:
マルチキャストグループ毎に特定のコアルータにパケットをいったん集約。ここまでは、(S, G) エントリによる経路管理。

下流のルータは、必要に応じてコアルータに参加要求を出し、経路を確定。コアルータ以下は、(*, G) エントリによる経路管理。

IPマルチキャスト (3)

- **DVMRP version 3**

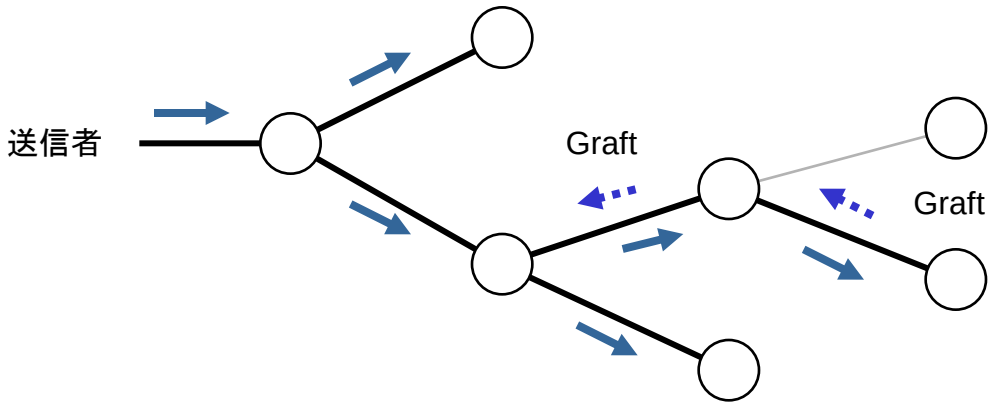
Prune メッセージ



Prune (刈り取り):
下流にマルチキャストグループ参加者がいない場合、上流ルータにパケット配送停止を要求。

途中のルータ: (S, G) エントリ削除。

Graft メッセージ



Graft (接ぎ木):
下流にマルチキャストグループ参加者が現れた場合、上流ルータにパケット配送再開を要求。

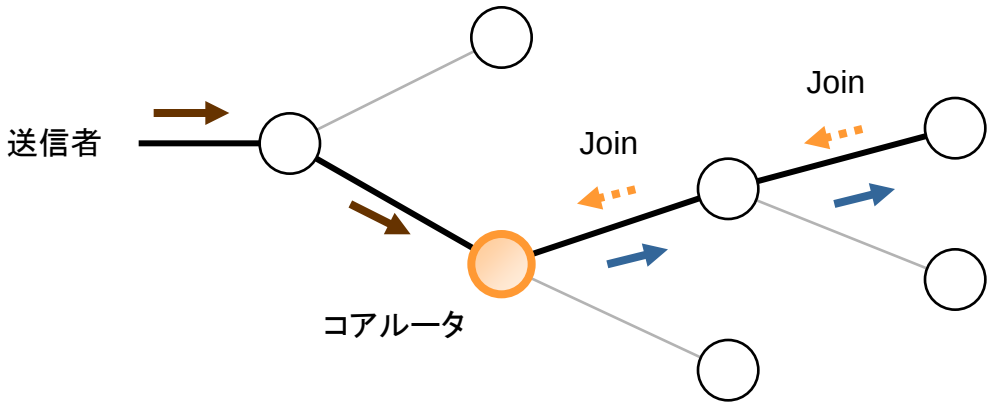
途中のルータ: (S, G) エントリ追加。

Distance Vector Multicast Routing Protocol

IPマルチキャスト (4)

- **PIM-SM**

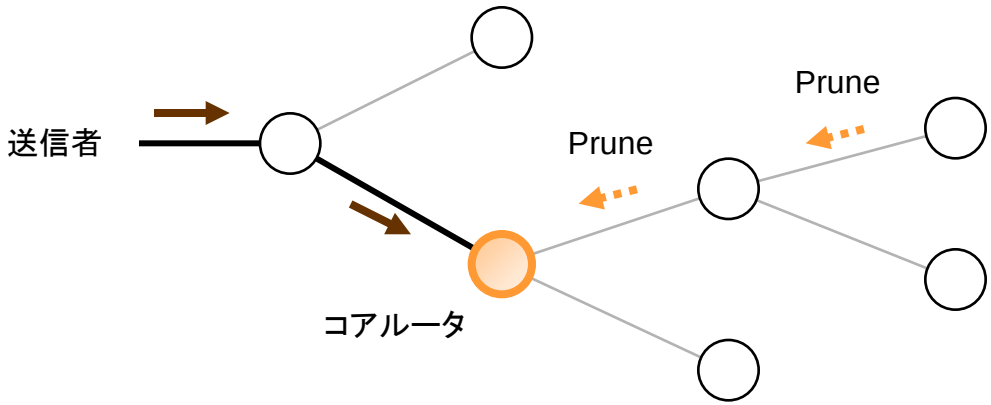
Join メッセージ



Join (参加):
下流にマルチキャストグループ参加者が現れた場合、上流ルータにパケット配送開始を要求。

途中のルータ: (*, G) エントリ追加。

Prune メッセージ



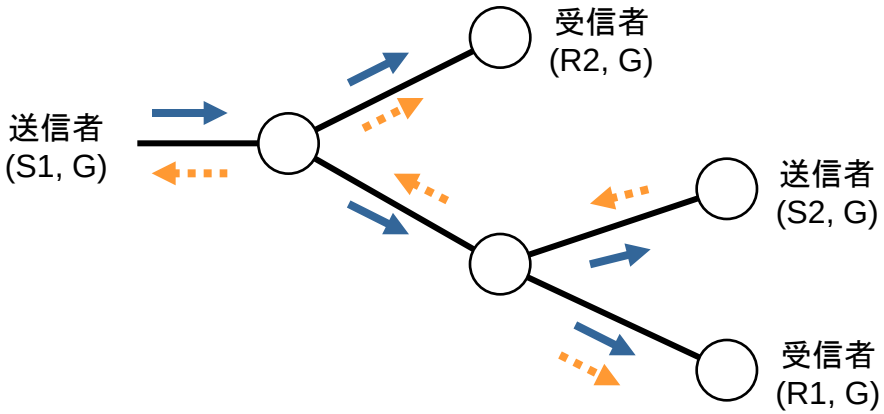
Prune (離脱):
下流のマルチキャストグループ参加者が離脱した場合、上流ルータにパケット配送停止を要求

途中のルータ: (*, G) エントリ削除

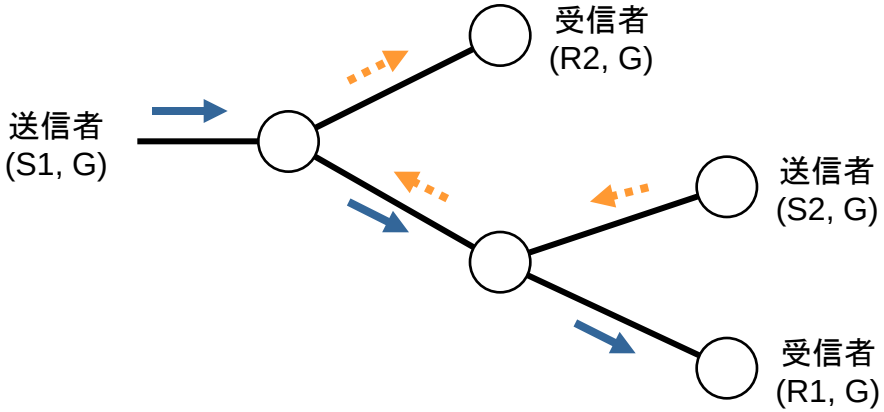
IPマルチキャスト (5)

- **SSM**

Any Source



Source Specific



ASM (Any Source Multicast: 従来)
同じマルチキャストアドレス G を使用するセッションのすべての参加者にパケット配信

⇒ 同じマルチキャストグループに複数の送信者が送信可能 (many-to-many)
⇒ 多人数会議

SSM:
送信者によって限定される (S, G) セッション参加者のみにパケット配信

⇒ 送信者を一人に限定 (one-to-many)
⇒ インターネット放送
(232.0.0.0 ～ 232.255.255.255)

Source Specific Multicast

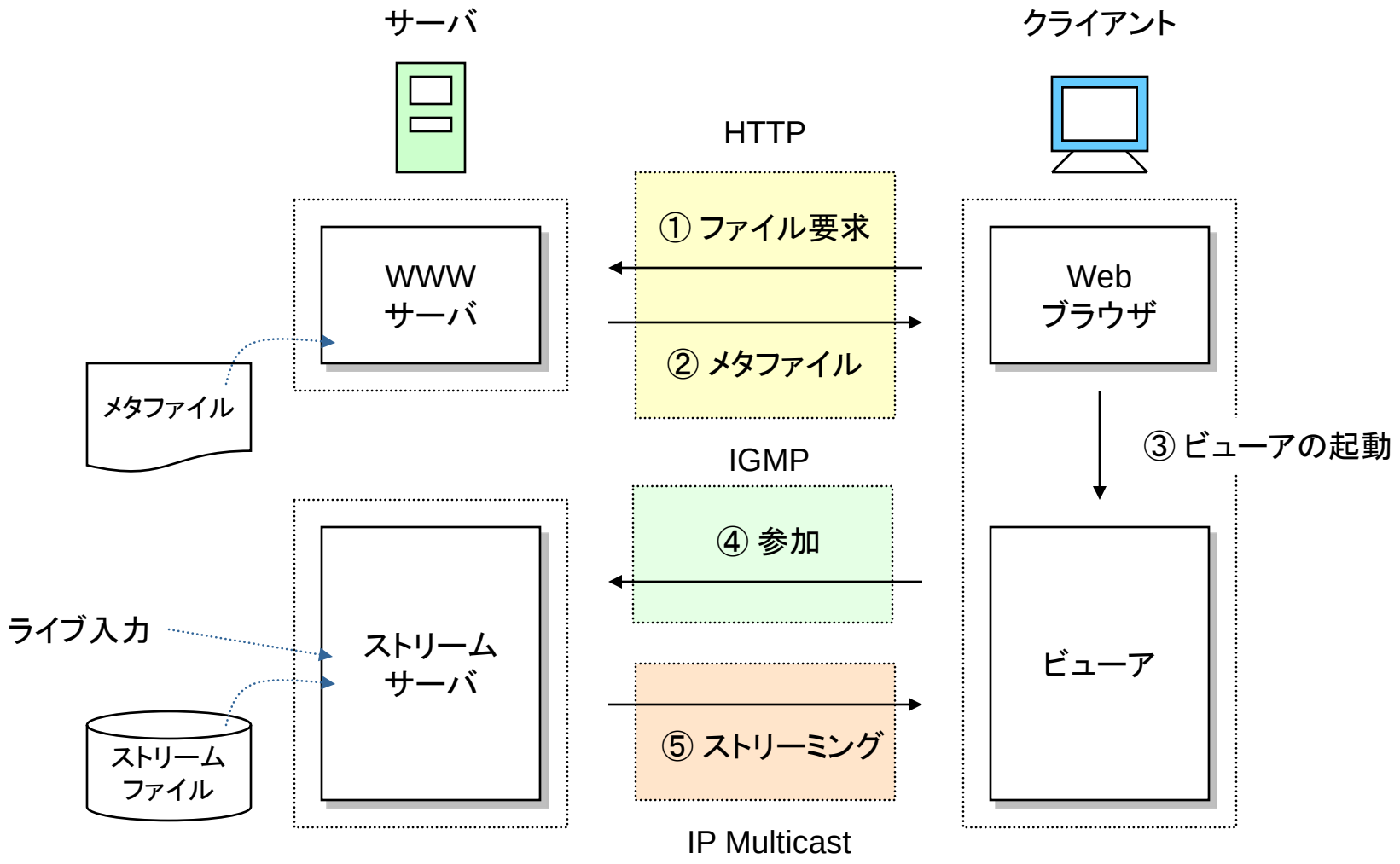
IPマルチキャスト (6)

- まとめ

プロトコル名	特徴	長所	短所
DVMRP	最小経路 (S, G) 送信者がパケットを投げると、フラッディングによって最小経路を確定、配信	最小経路	フラッディングによる不要なトラヒックの増加 ⇒ 拡張性
PIM-SM	送信者・コアルータ: 最小経路 (S, G) コアルータ・受信者: 共有経路 (*, G) 送信者がコアルータに「登録」すると、最小経路を確定 受信者がコアルータに「参加」すると、共有経路を確定、配信	フラッディングが不要 ⇒ 拡張性	共有経路が必ずしも最短経路にならない コアルータの決定方法 プロトコルが若干複雑 (最短経路と共有経路の動的切替え)
SSM	最小経路 (S, G) 受信者が送信者に subscribe すると最小経路を確定、配信	1 対多の放送型アプリケーション PIM-SM とのハイブリッド構成 (PIM-SSM)	1 対多に限定 IGMP v3 が必須

マルチキャスト放送 (1)

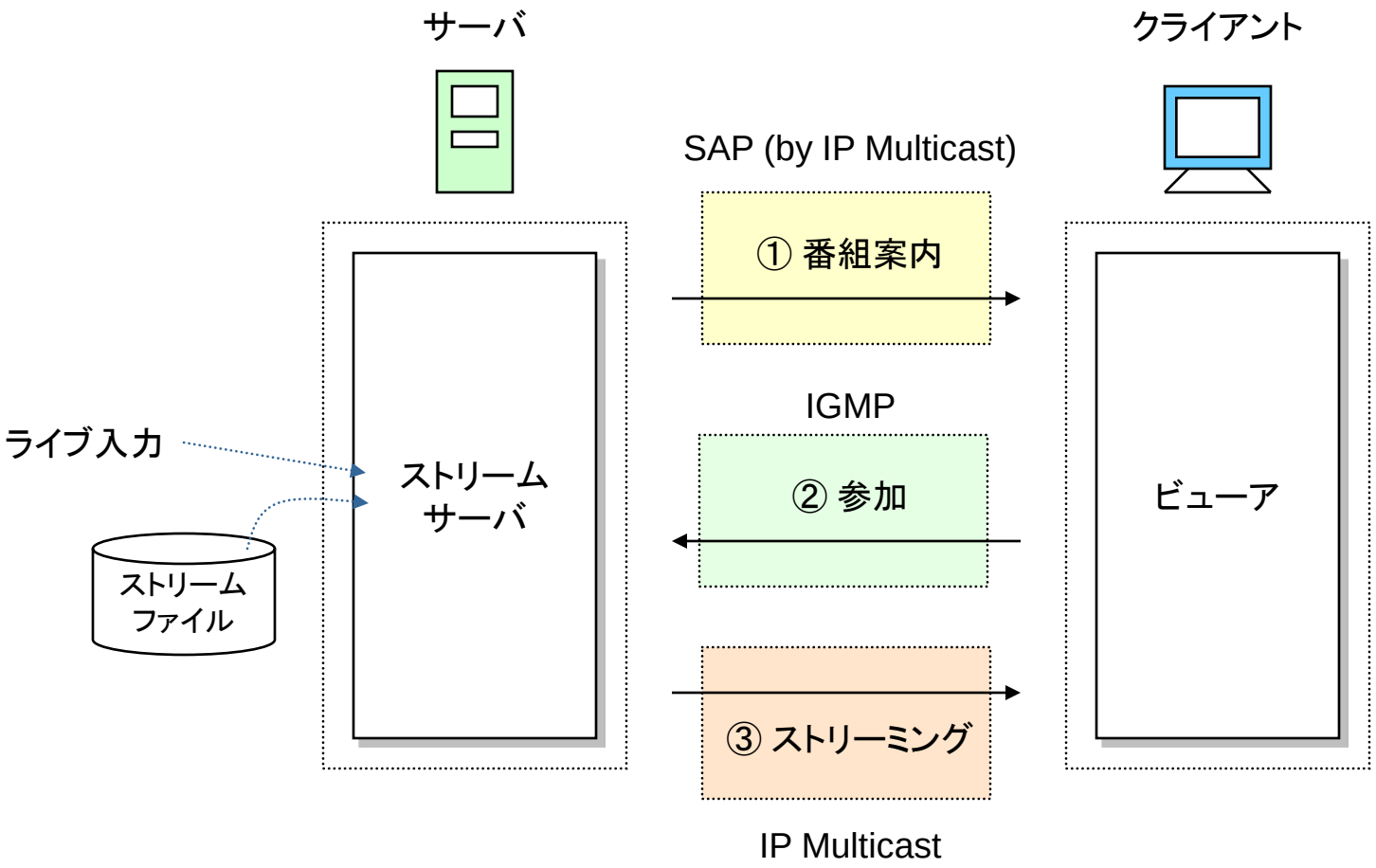
- (1) WWW による番組案内



マルチキャスト放送 (2)

- **(2) SAP による番組案内**

SAP: Session Announcement Protocol
定期的に番組案内 (SDP) をマルチキャスト



RFC 2974：vic/rat/sdr

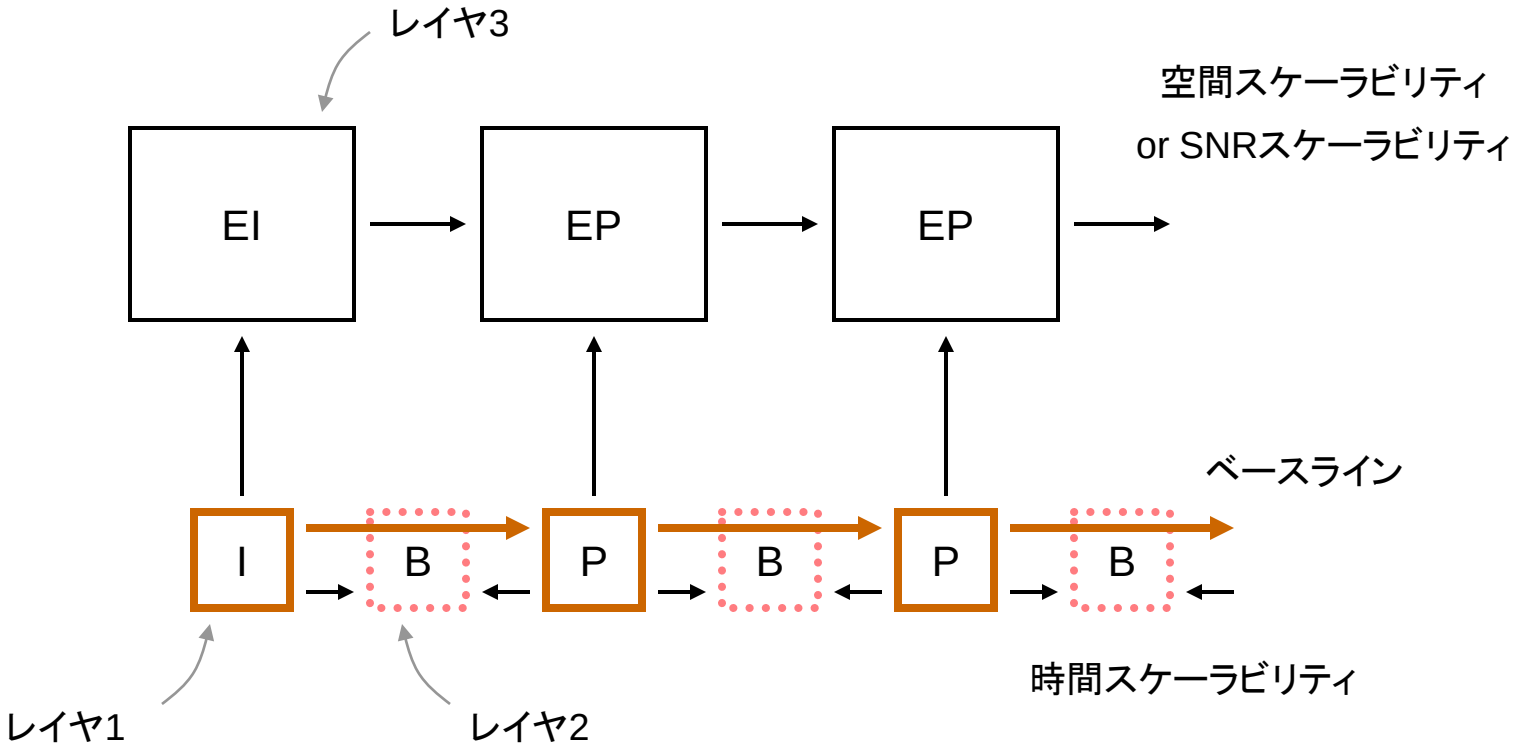
マルチキャスト放送の長所と短所

	ユニキャスト放送	マルチキャスト放送
長所	既存のシステムの変更が不要 クライアントの接続状況に合わせたふくそう制御が可能	トラヒックの削減 (原理的に冗長なパケットは発生しない) 、およびサーバ負荷の削減
短所	クライアントの増加に伴うトラヒックの爆発、ならびにサーバ負荷の増大 (線形増加)	マルチキャストルータの普及と各種設定 クライアント毎のふくそう制御が困難
課題		マルチキャストルーティングプロトコル ふくそう制御アルゴリズム

例: 階層化マルチキャスト

階層化マルチキャスト

スケーラブル符号化



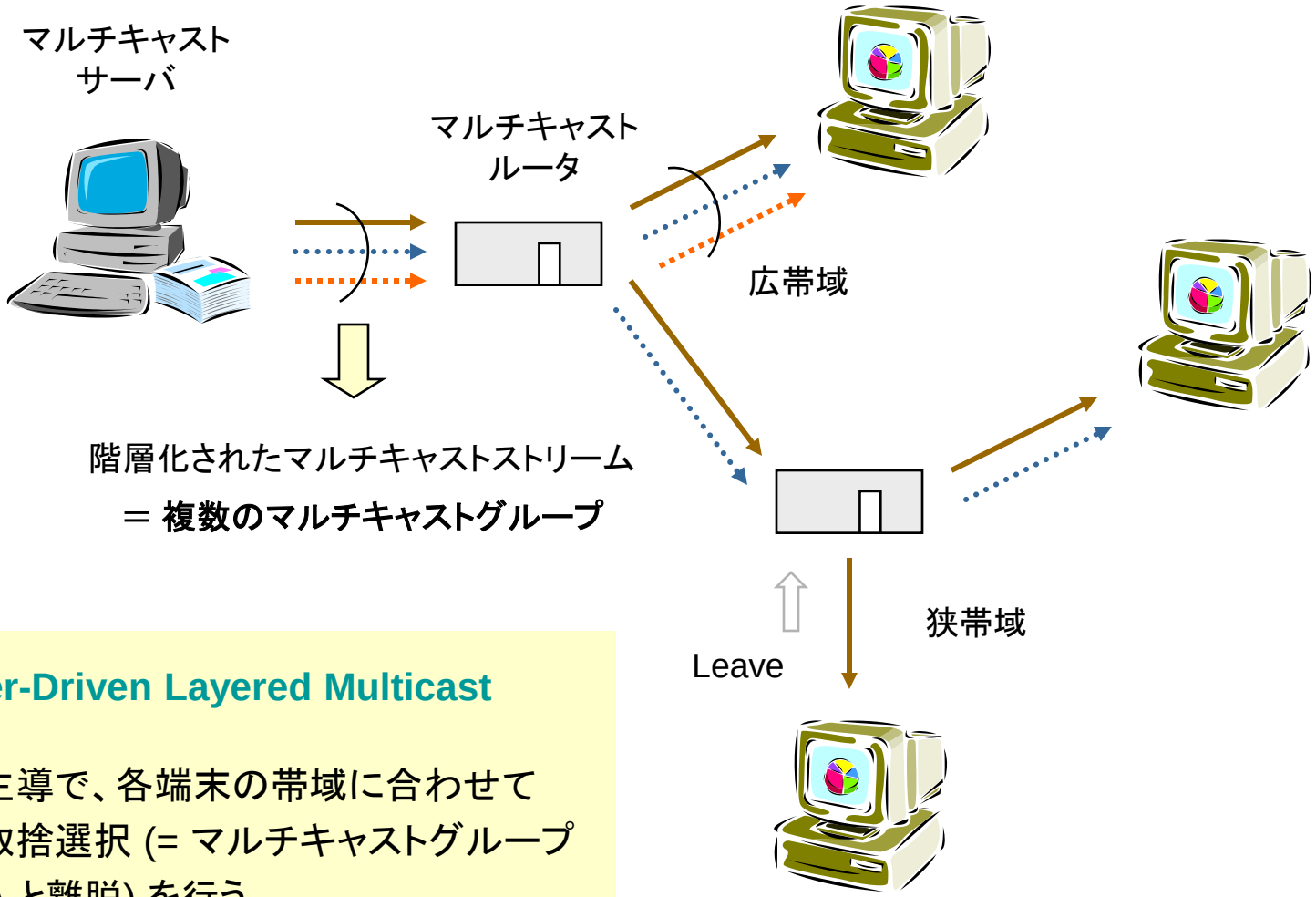
- 空間解像度の階層化：空間スケーラビリティ
- 時間解像度の階層化：時間スケーラビリティ
- SNRの階層化：SNRスケーラビリティ

レイヤ1のみ：低品質、低レート

〜

すべてのレイヤ：高品質、高レート

階層化マルチキャスト (1)



Receiver-Driven Layered Multicast

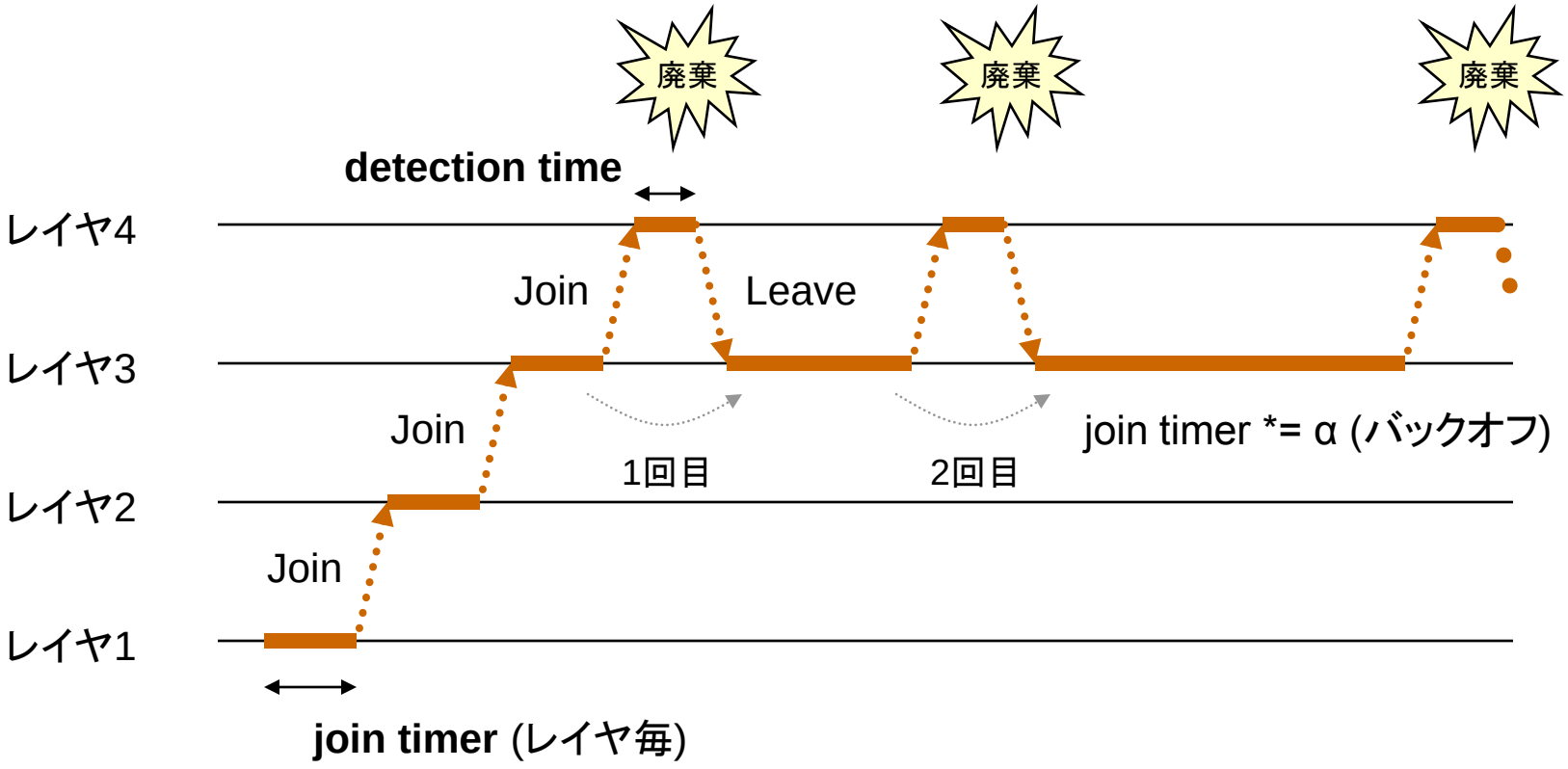
受信者主導で、各端末の帯域に合わせて階層の取捨選択 (= マルチキャストグループへの加入と離脱) を行う

S.MaCanne et al: “Receiver-driven Layered Multicast,” SIGCOMM’96.

階層化マルチキャスト (2)

- **Join Experiment**

Join、Leave (ふくそう検出)、バックオフを繰り返し、レートを安定させる

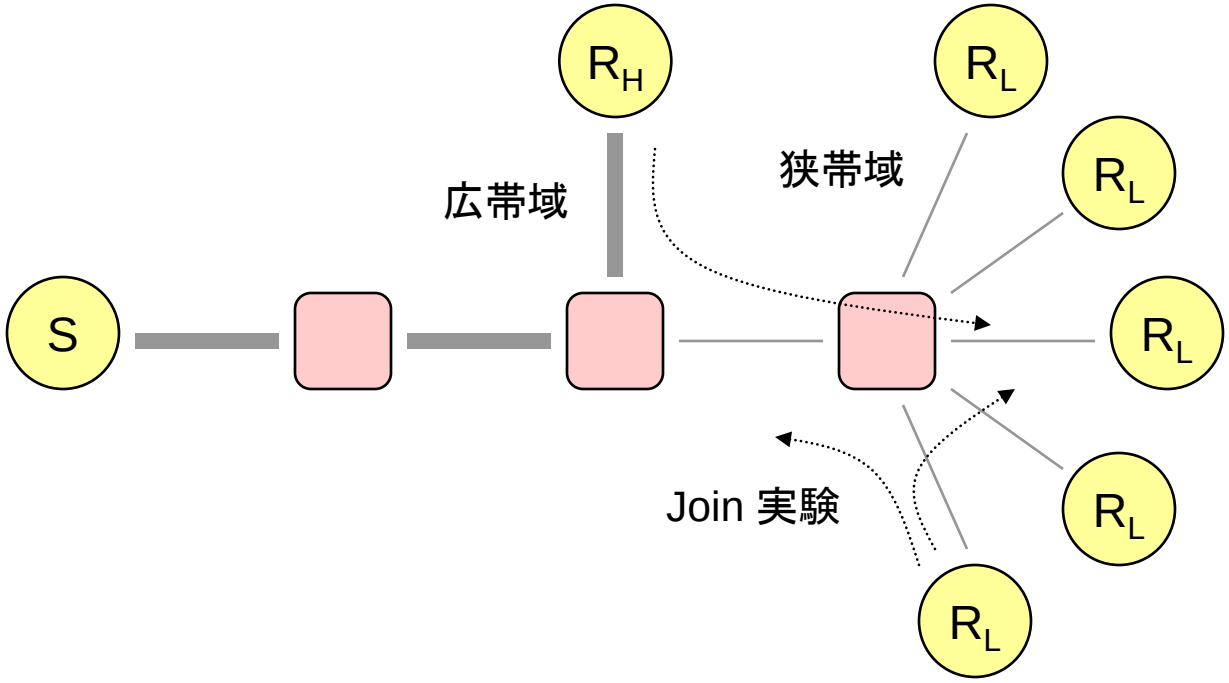


TCP タイムアウトと同様のバックオフメカニズム

階層化マルチキャスト (3)

- **Shared Learning**

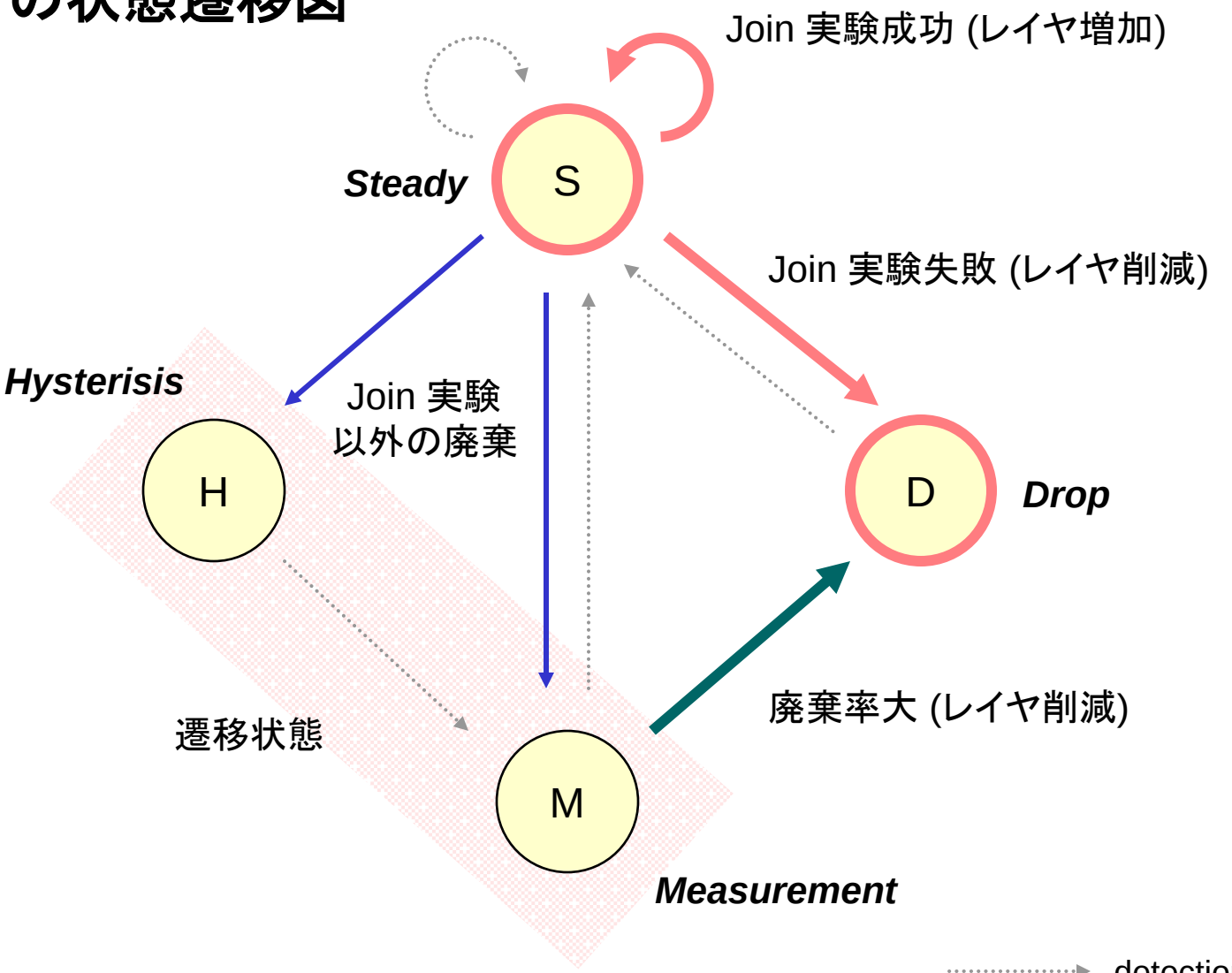
Join 実験の他の端末への通知



- 端末数の増加に伴う Join 実験の回数の増加を防ぐ
- 上流の広帯域 Join 実験と下流の狭帯域 Join 実験の結果の混同を防ぐ

階層化マルチキャスト (4)

- **RLM の状態遷移図**

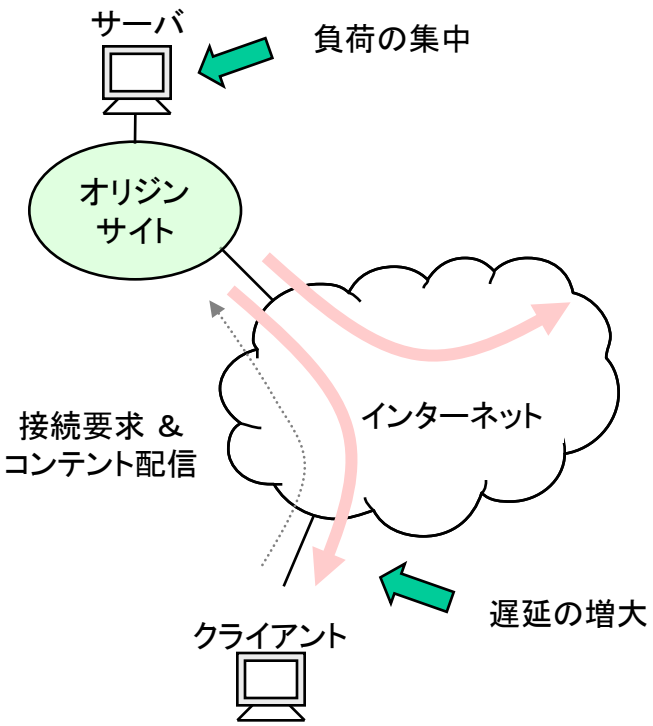


CDN

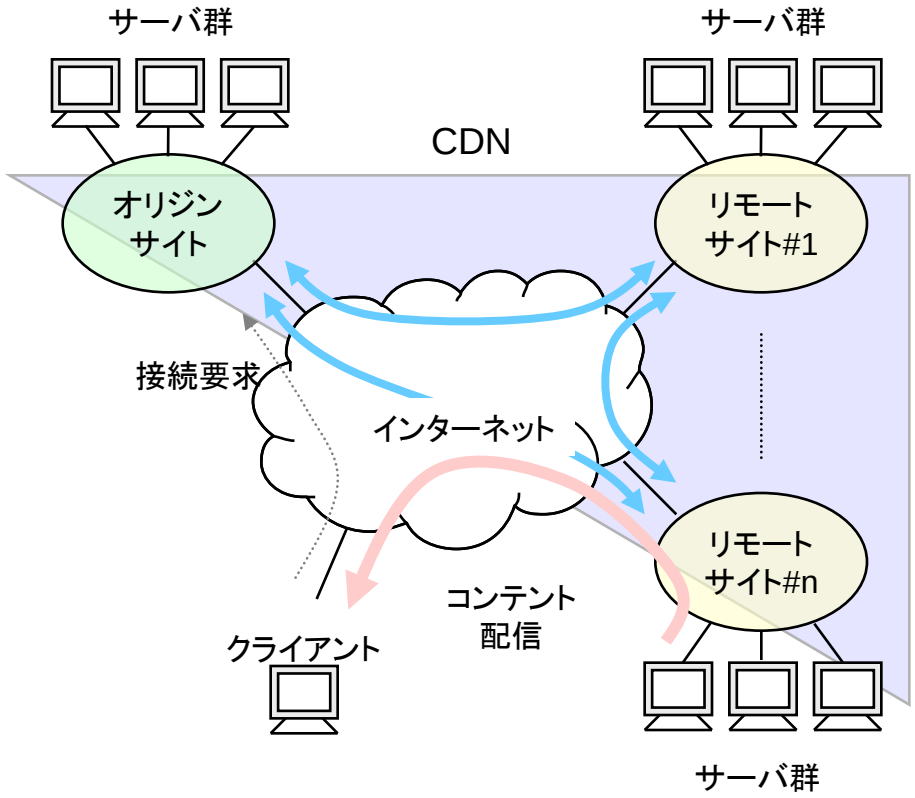
Content Delivery Network

CDN

- サーバの負荷分散 & 転送遅延の改善

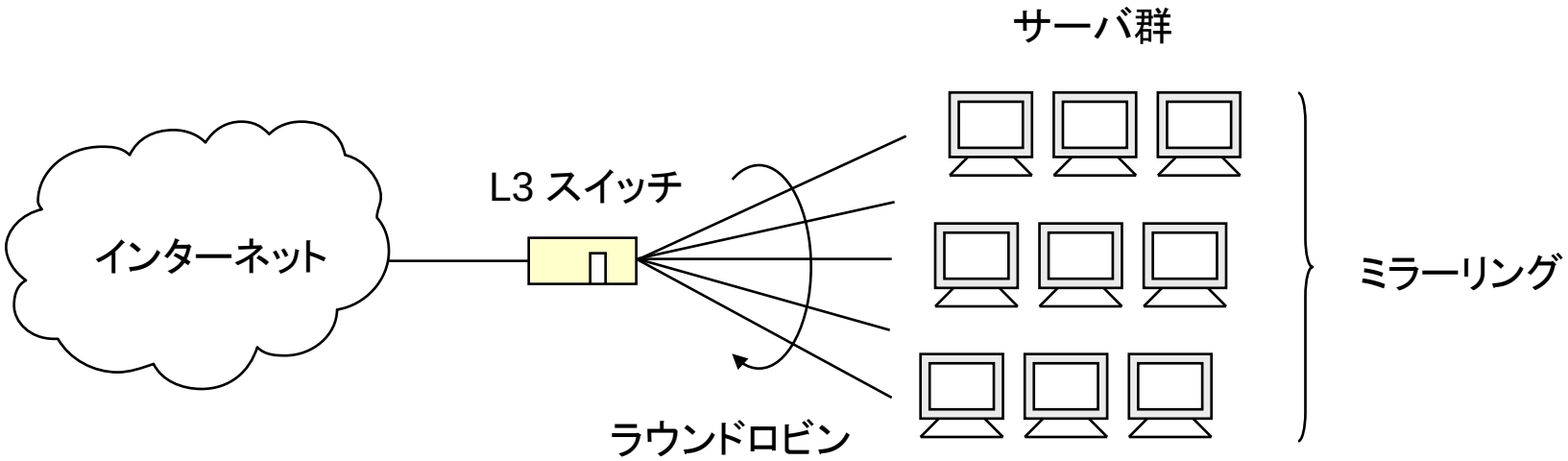


- 複数サーバによるサイト内負荷分散
- 複数サイトによる負荷分散・遅延改善



サイト内負荷分散 (1)

- **L3 スイッチ**



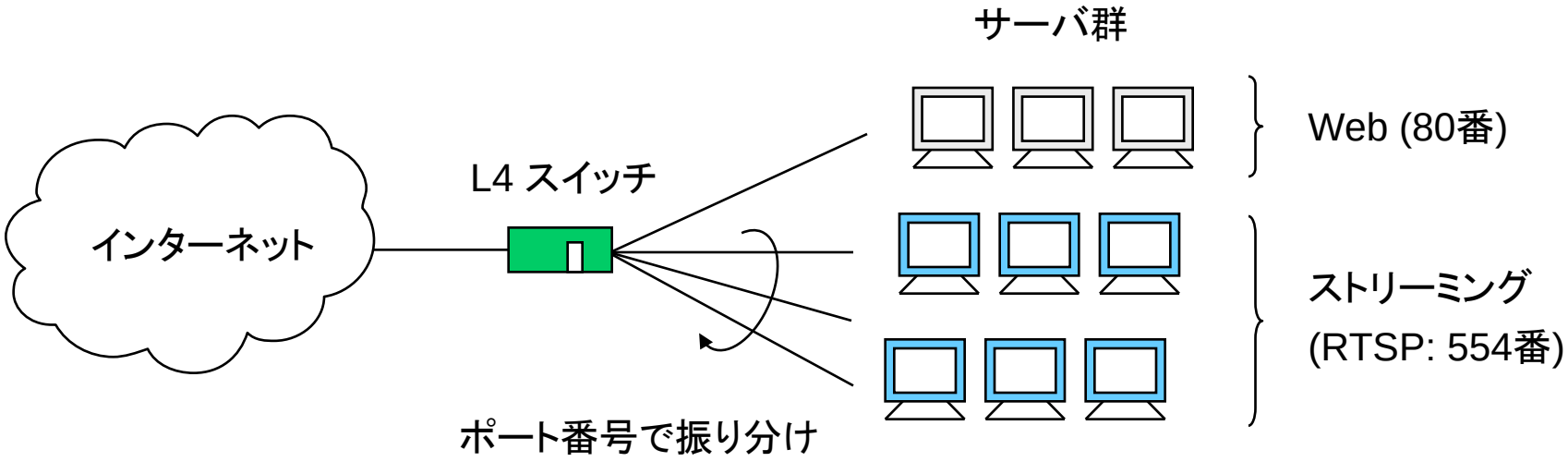
ミラーリングとラウンドロビンによる負荷分散:

長所: スイッチの負荷が軽い

短所: ミラーリングの効率が悪い (すべてのサーバが同じデータを持つ)

サイト内負荷分散 (2)

- **L4 スイッチ**



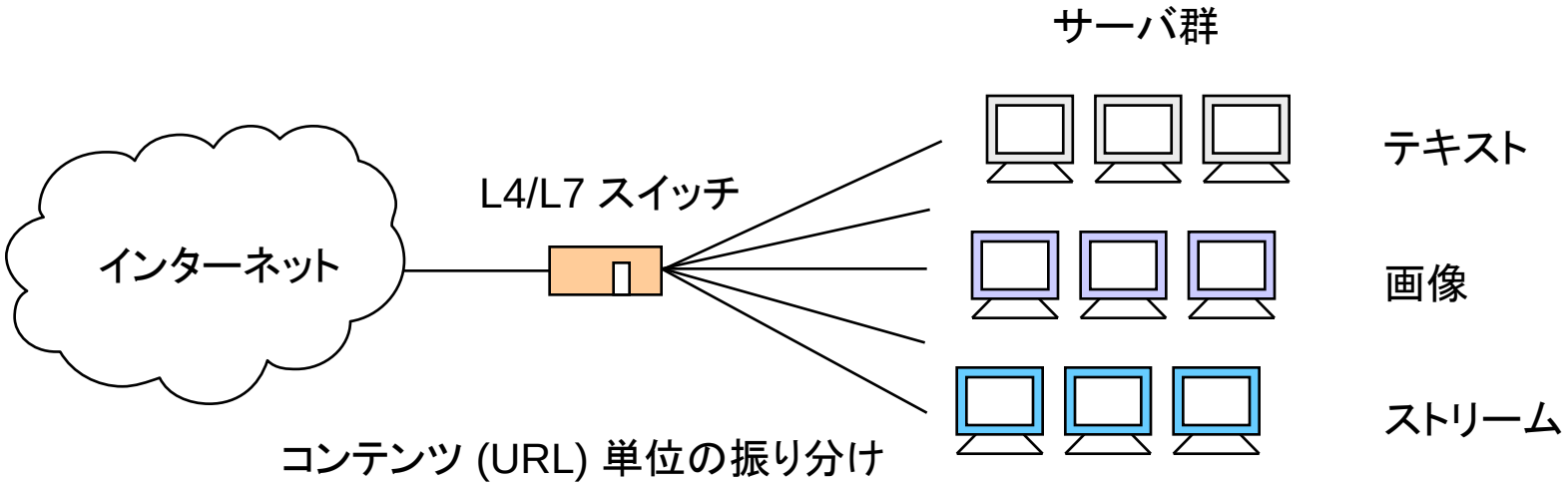
アプリケーション (ポート番号: L4情報) に応じた分散サーバ配置:

長所: アプリケーションに応じたきめこまかい負荷分散が可能

(短所: L3 スイッチよりはスイッチの負荷が大きい)

サイト内負荷分散 (3)

- **L4/L7 スイッチ**



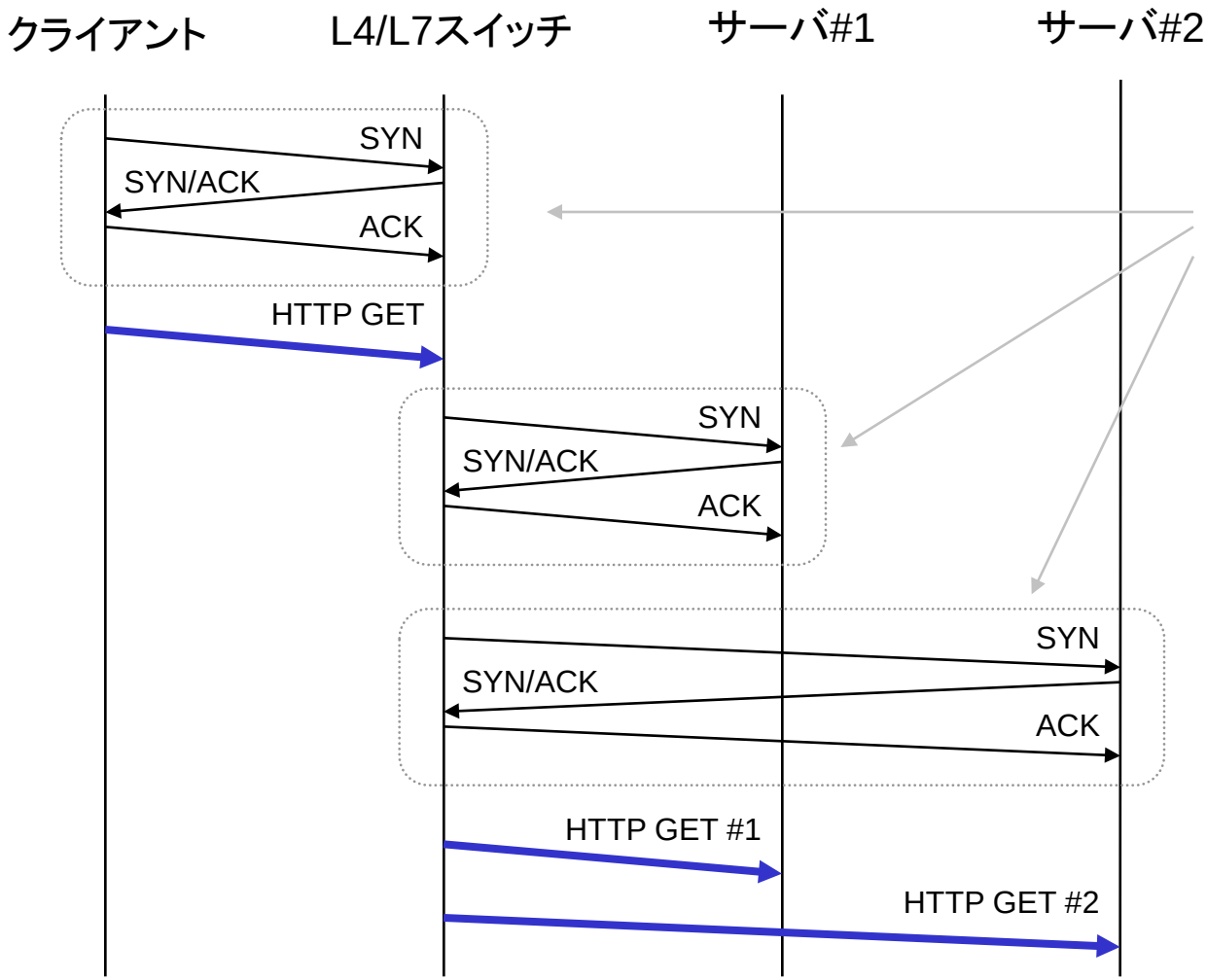
コンテンツ (URL: L7情報) に応じた分散サーバ配置:

長所: コンテンツ単位のさらにきめこまかい負荷分散が可能

短所: スイッチの負荷が大きい

サイト内負荷分散 (4)

- **Delayed Bound (1)**

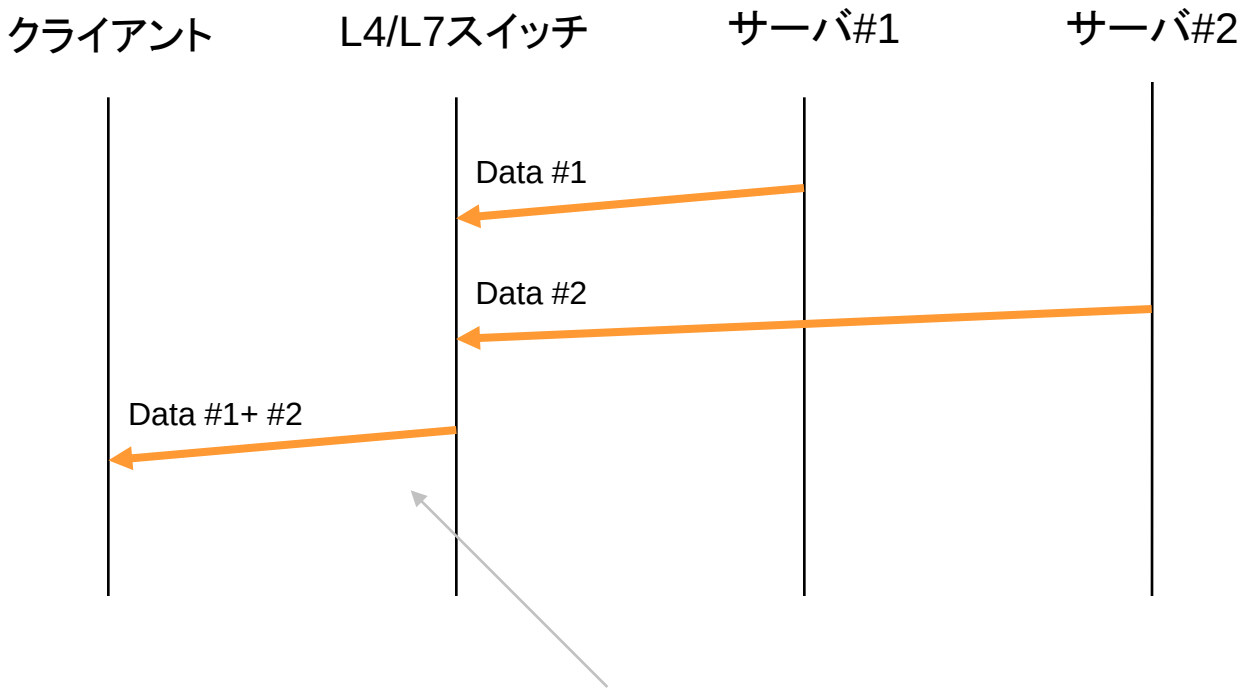


クライアント・スイッチ間、
スイッチ・サーバ間で
複数の TCP コネクション
を終端
＝ Delayed Bound

HTTP 1.1 の例

サイト内負荷分散 (5)

- **Delayed Bound (2)**

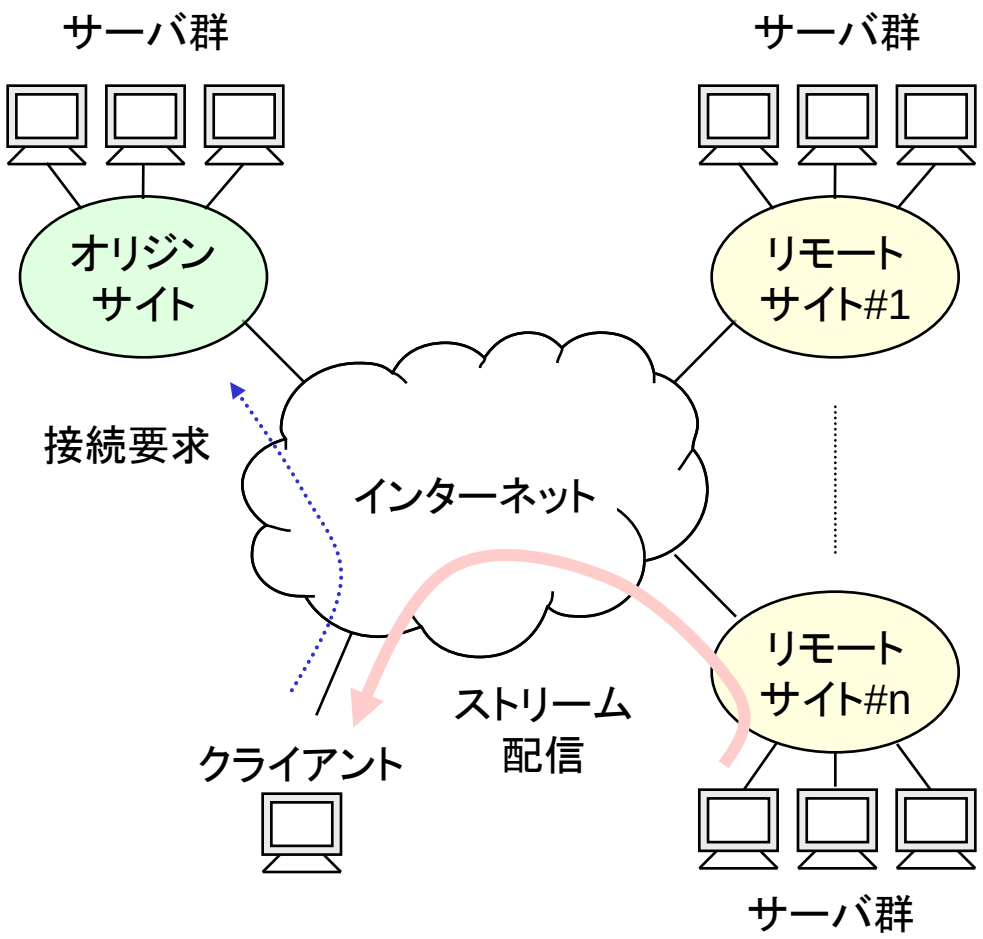


サーバ#1、サーバ#2
からのデータを集約
= Aggregate

HTTP 1.1 の例

サイト間負荷分散

- サイト間負荷分散 & 転送遅延の改善



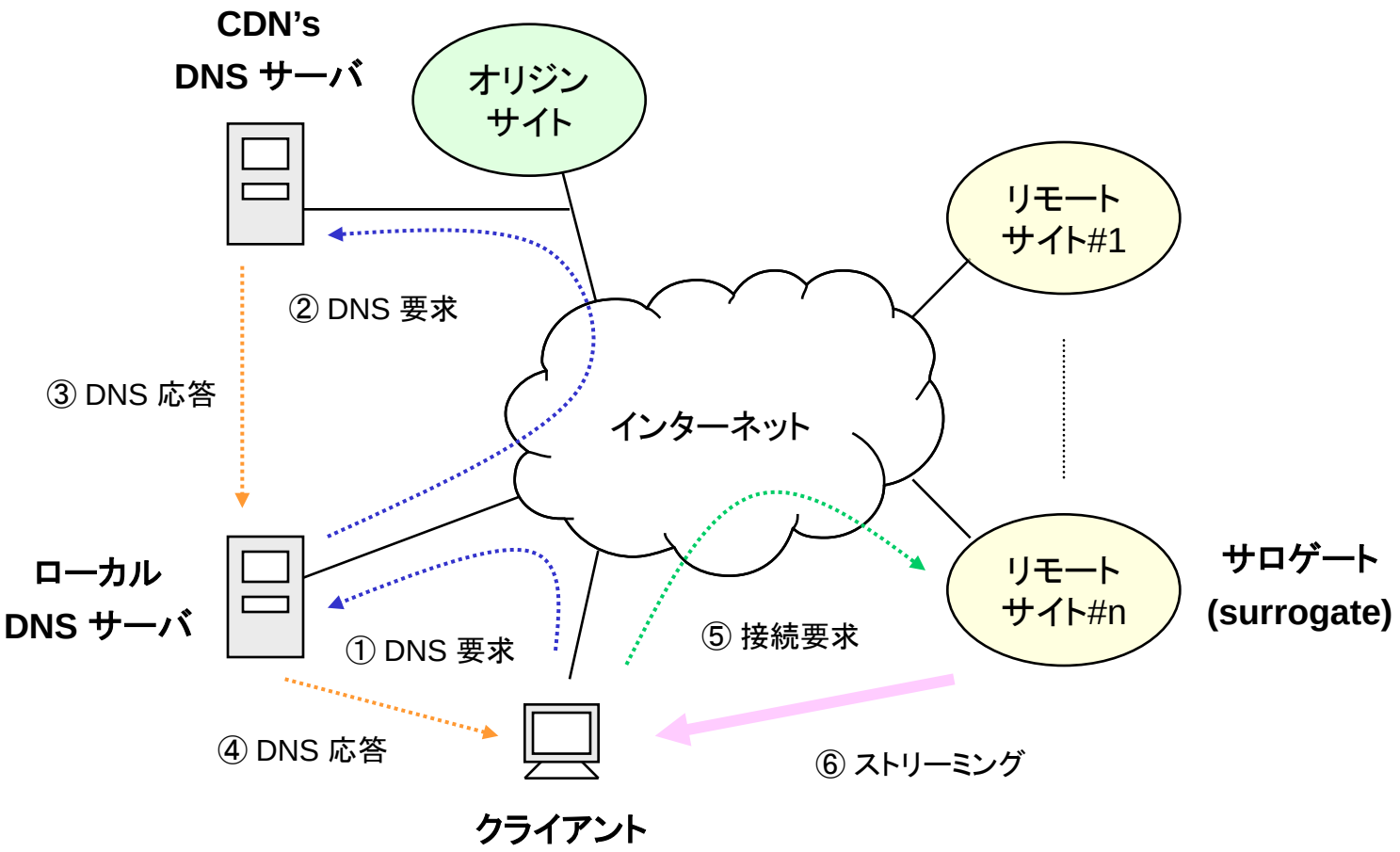
複数サイト (サーバ群)
の分散配置

クライアントからの要求
に応じて、適切なサイト
を選択、誘導

サイト間負荷分散 &
転送遅延の改善

リクエストルーティング (1)

- **DNS リダイレクション (1)**



解像度: ドメイン単位 (粗い)

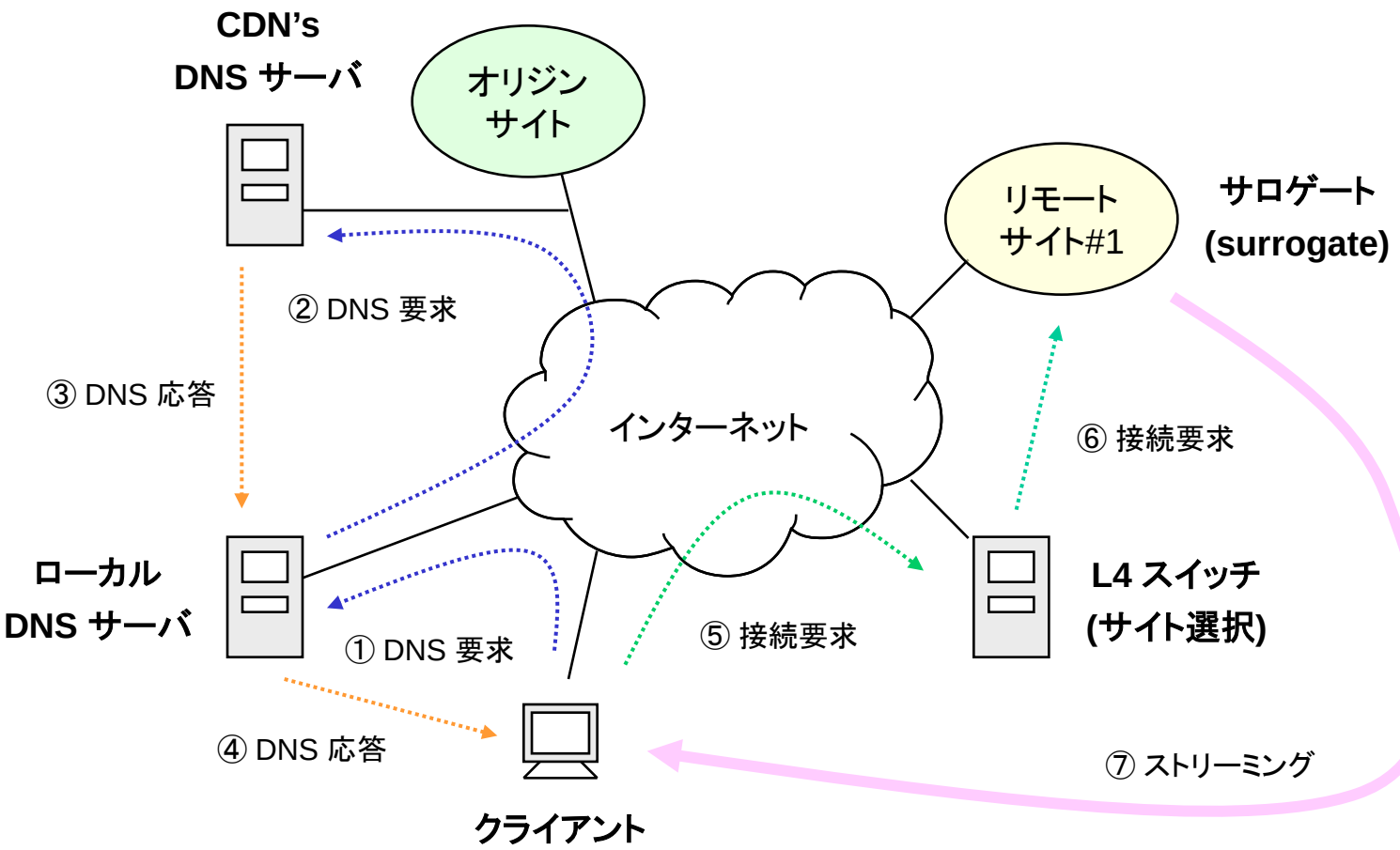
リクエストルーティング (2)

- DNS リダイレクション (2)

DNS リダイレクション	方式
Single Reply	CDN 内 DNS サーバが最適サロゲートを A レコード（IP アドレス）で返す方式 （例： stream.com → 192.168.0.1）
Multiple Reply	CDN 内 DNS サーバが複数のサロゲート候補を A レコードで返し、ラウンドロビンでサロゲートを選択する方式 （例： stream.com → 192.168.0.1, 192.168.0.2, 192.168.0.3 → 192.168.0.2）
NS Redirection	CDN 内 DNS サーバが、第三の DNS サーバに NS レコード（ネームサーバ）を返し、その DNS サーバが最適サロゲートを A レコードで返す方式 （例： stream.com → server1.site1.stream.com → 192.168.0.3）
CNAME Redirection	CDN 内 DNS サーバが、第三の DNS サーバに CNAME レコード（エイリアス）を返し、その DNS サーバが最適サロゲートを A レコードで返す方式 （例： stream.com → site1.stream.com → 192.168.0.4）
Object Encoding	DNS の名前にオブジェクトのタイプ等を埋め込んでしまい、それに応じてサロゲートの IP アドレスを振り分ける方式 （例： stream.com → mpeg_content1.site1.stream.com → 192.168.0.5）

リクエストルーティング (3)

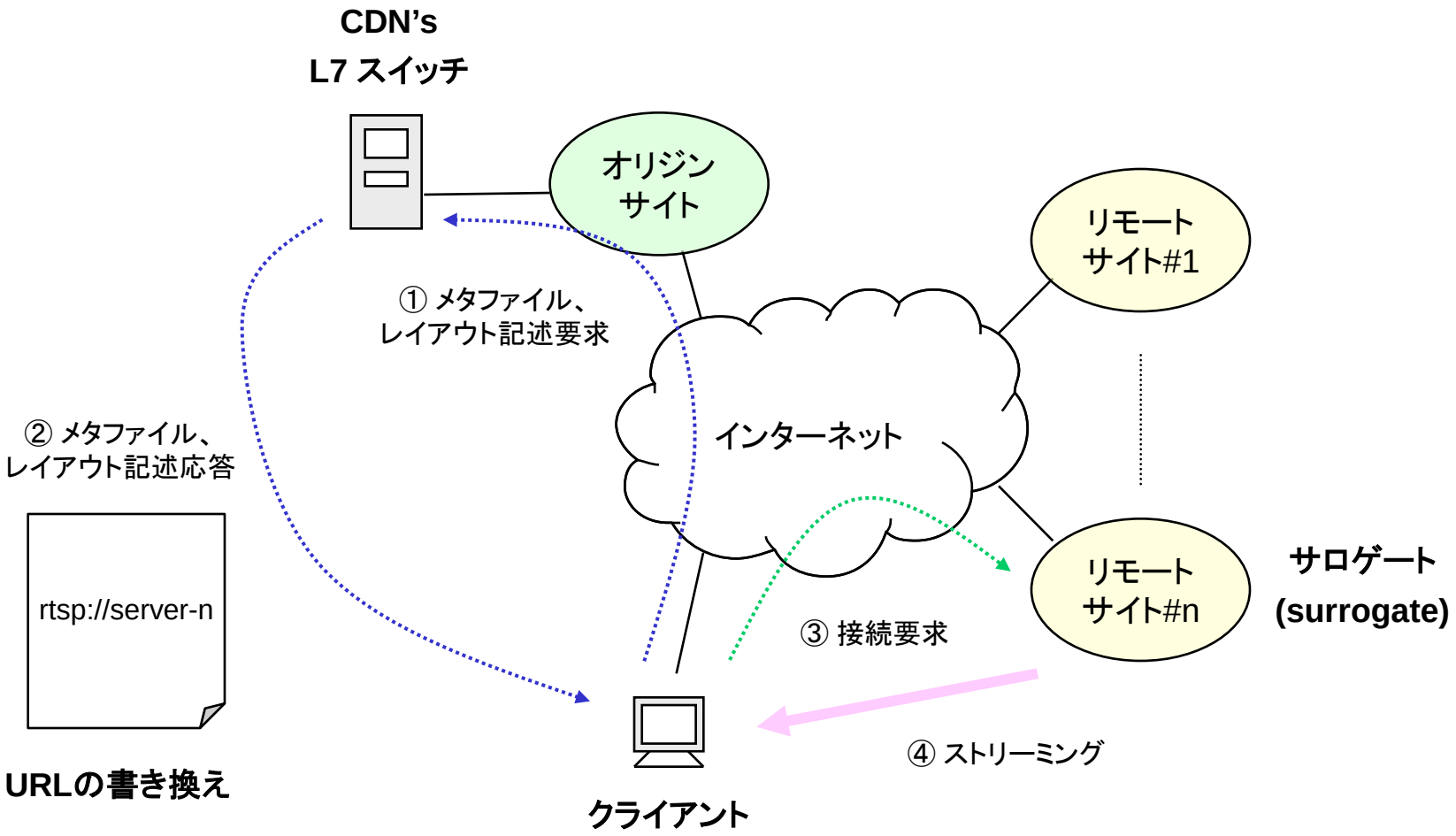
- DNS リダイレクション + L4 スイッチ



サロゲートの IP アドレスを返す代わりに L4 スイッチの IP アドレスを返す (負荷分散)

リクエストルーティング (4)

- URL リライティング (L7 スイッチ)



解像度: クライアント単位 (細かい)

リクエストルーティング (5)

- URL リライティング (2)

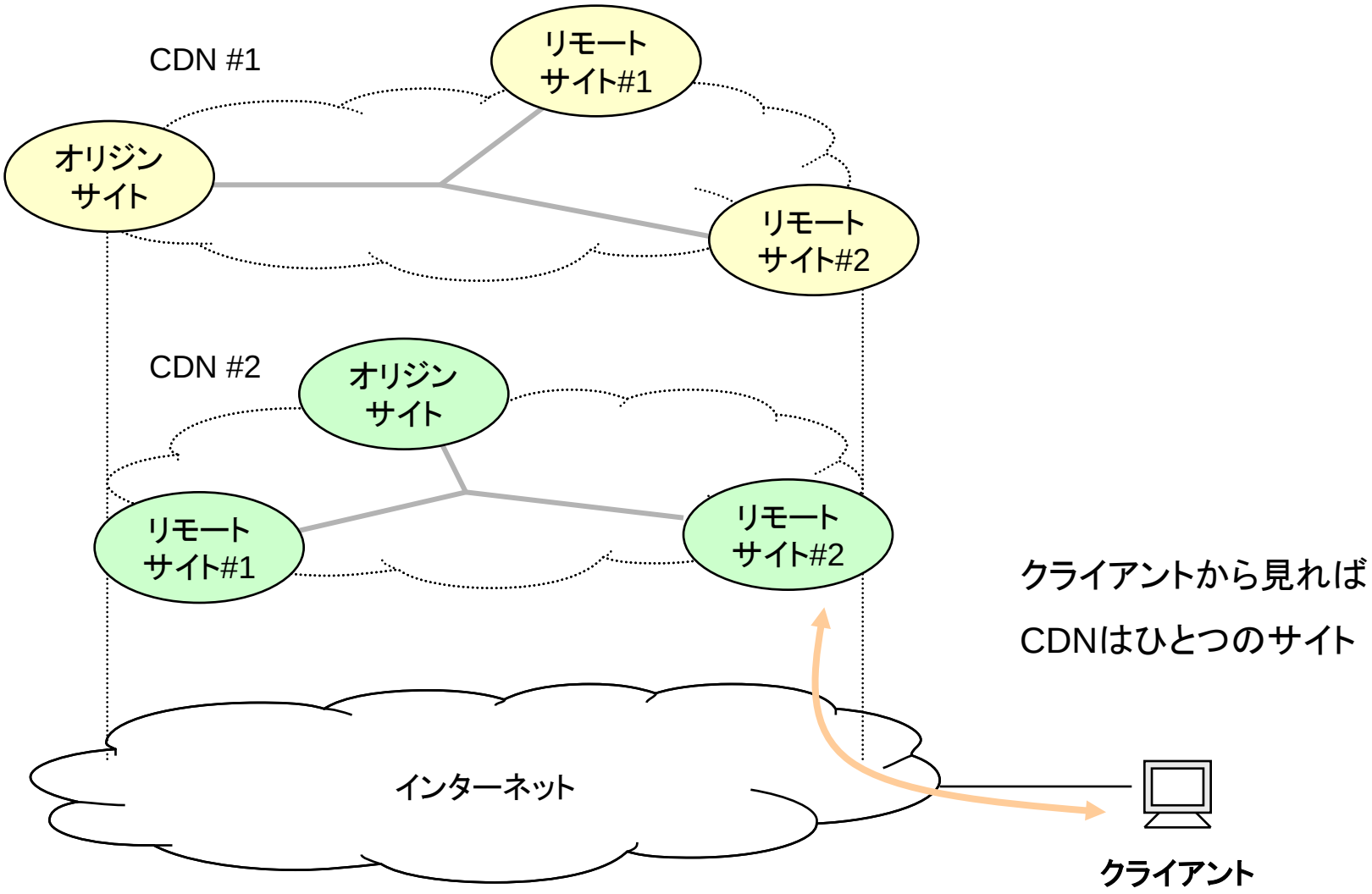
URL リライティング	方式
Header Inspection (1)	RTSP 記述内に仮想的なサロゲートの URL を記述しておき、アクセスが来たら最適サロゲートへの 302 リダイレクションコードを返す (例) “302” Moved Temporarily
Header Inspection (2)	MIME ヘッダ内の Language、Cookie 等のフィールド情報に応じて、適切なサロゲートへのルーティングを行う (例) stream.com → japanese.stream.com
Content Modification	クライアントからのリクエストに応じて、メタファイルやレイアウト記述ファイル内の URL フィールドを最適サロゲートの URL に書き換えて返す (例) rtsp://stream.com → rtsp://site1.stream.com

リクエストルーティング (6)

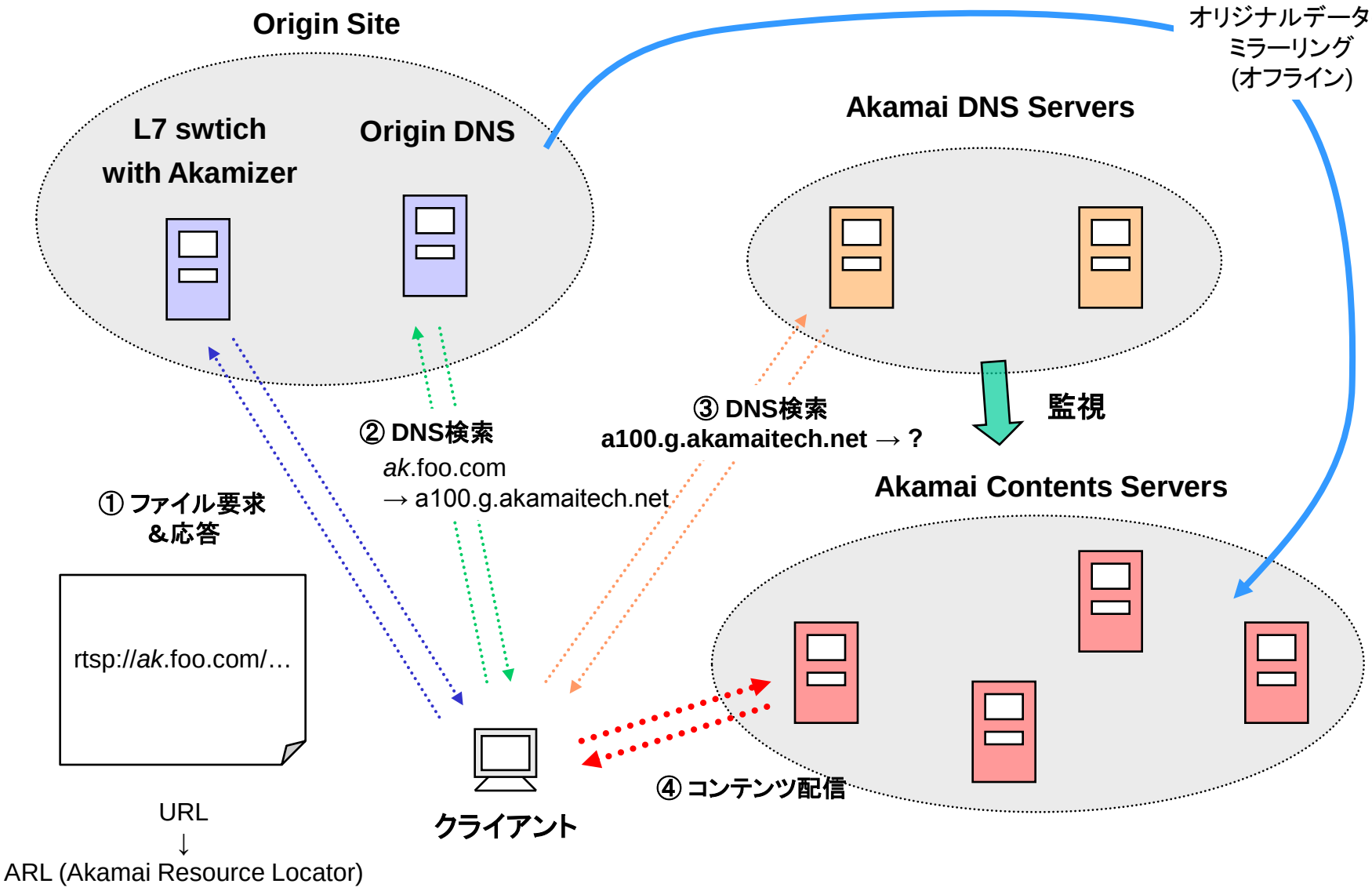
- 最適サロゲートの推定方法

推定方法	方式
Proximity Measurement	クライアントに最も近いサロゲートの推定方法 (1) Active Probing : ping 等のプローブパケットの利用 (2) Passive Measurement : クライアントパケットのモニタリング 基準: 遅延、パケットロス、ホップ数、等 関連分野: インターネットの帯域測定技術
Surrogate Feedback	管理サーバとサロゲートの情報交換: エージェントを用いた Probing 基準: CPU 負荷、インターフェース負荷、コネクション数、等 関連分野: 負荷分散技術

オーバーレイネットワーク



Akamai FreeFlow (1)



① URLリライテイング、③ DNSリダイレクション

Akamai FreeFlow (2)

- **Akamai DNS System**

High-Level DNS Servers (世界中に13台？)

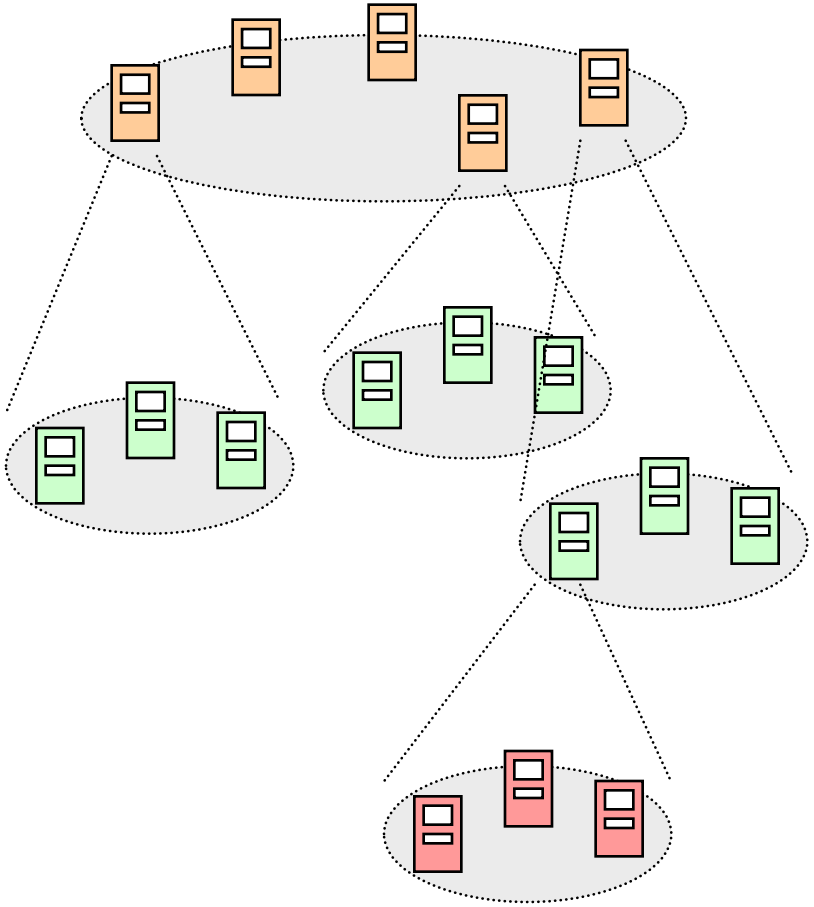
za.akamaitech.net
zb.akamaitech.net
…
zr.akamaitech.net

Low-Level DNS Servers (50以上)

n1g.akamaitech.net
n2g.akamaitech.net
…
n9g.akamaitech.net

Contents Servers (2000以上)

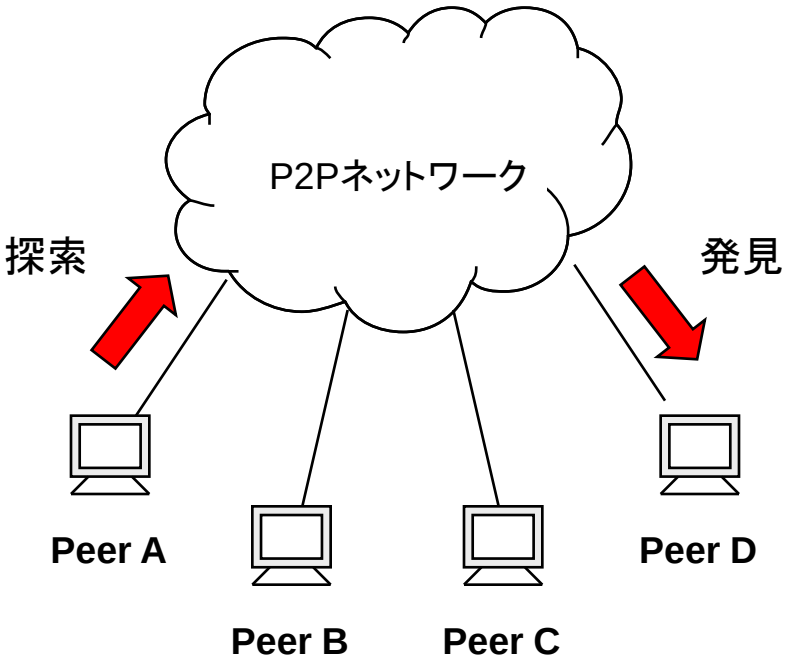
a0000.g.akamaitech.net
a0001.g.akamaitech.net
…
a*nnnn*.g.akamaitech.net



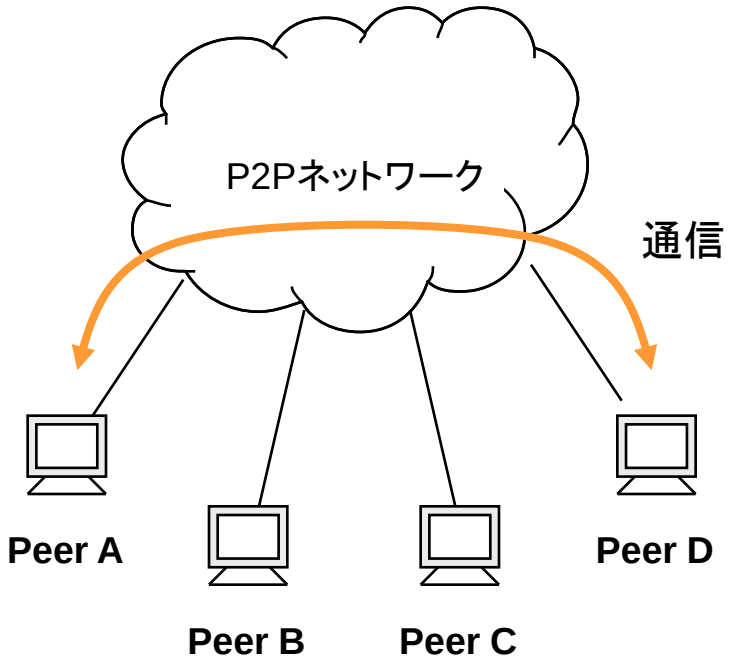
P2P
(peer-to-peer)

P2P (1) 基本

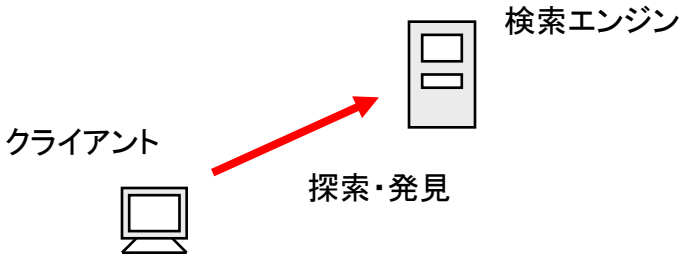
(1) 探索・発見

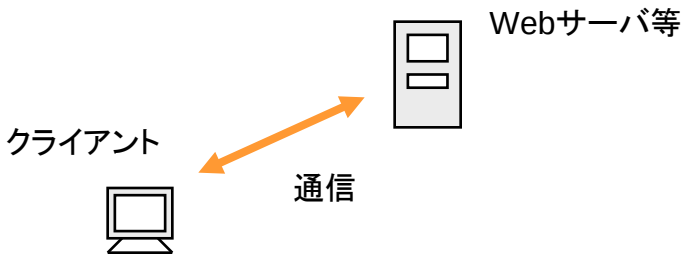


(2) 通信

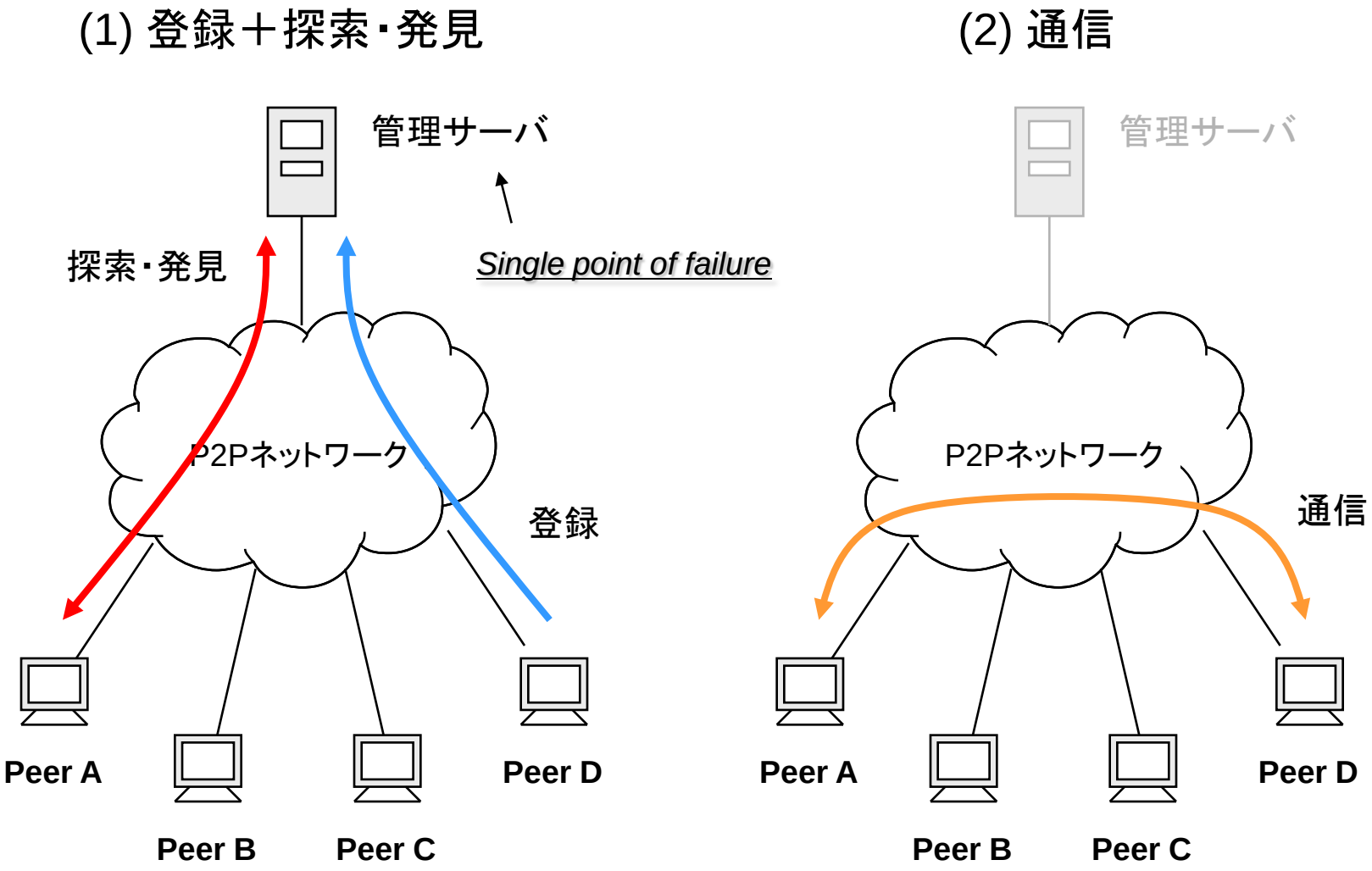


従来：



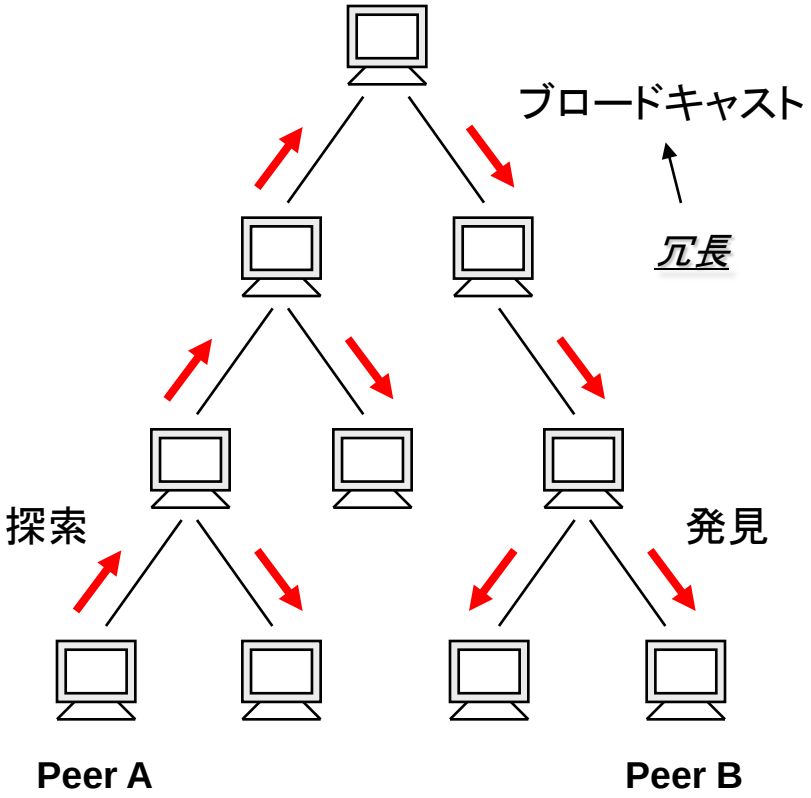


P2P (2) Napster

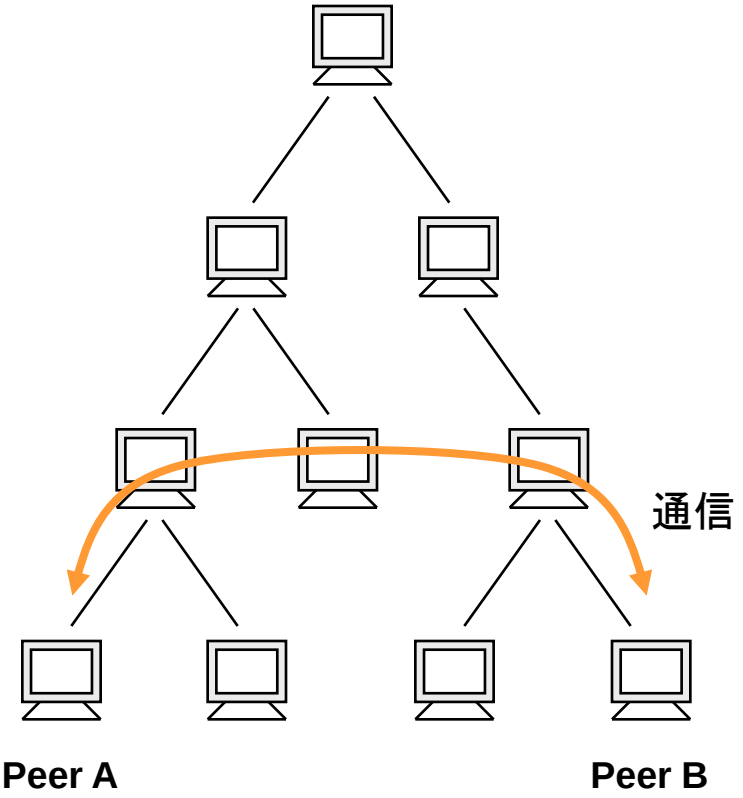


P2P (3) Gnutella

(1) 探索・発見



(2) 通信

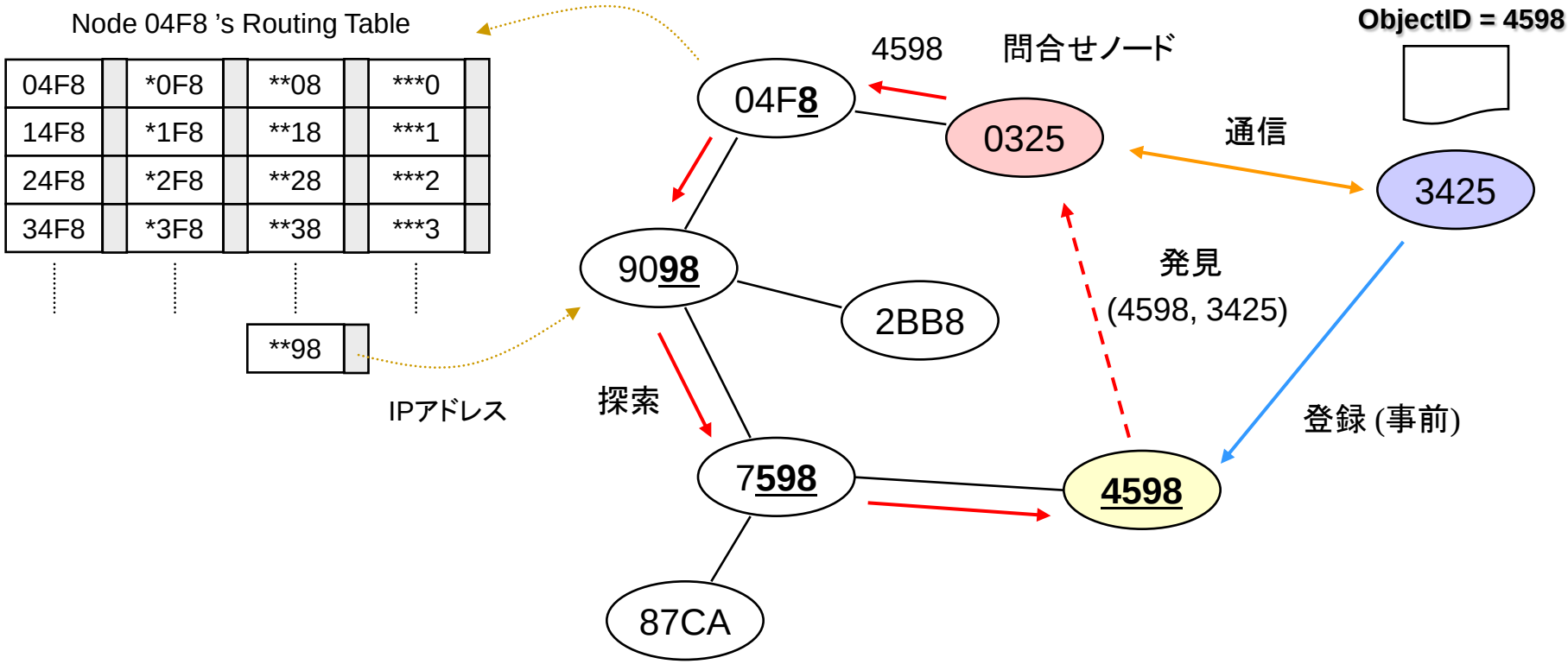


P2P (4) Plaxton's Algorithm

ファイル名とノードアドレスをハッシュ関数で数値化 … (ObjectID, NodeID)

ノード番号が ObjectID に等しいノードに、そのファイルの保有ノード情報を登録

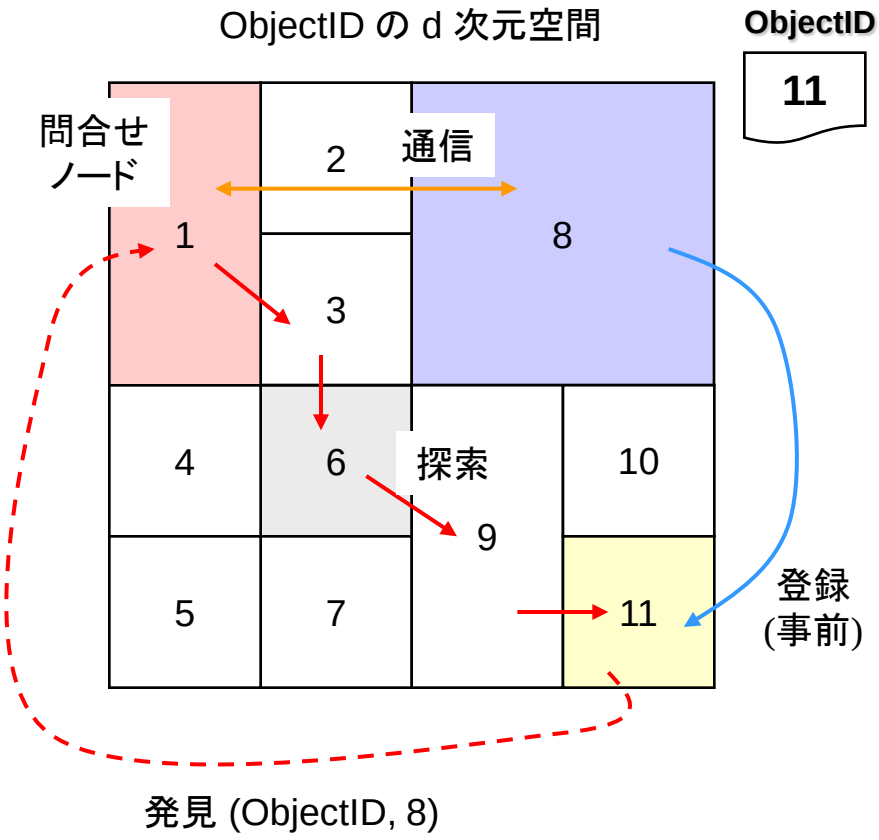
探索・発見： ***8 ⇒ **98 ⇒ *598 ⇒ 4598 の順に探索 (ObjectID = 4598 の場合)



P2P (5) CAN

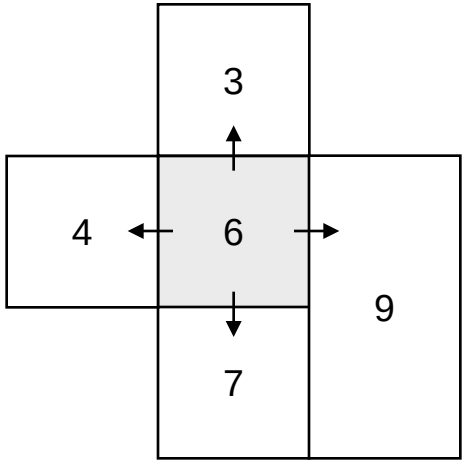
Plaxton's Algorithm の変形、拡張

各ノードは、d 次元空間中の特定の範囲の ObjectID を有するファイルの保有ノード情報を保持



(例) ノード6におけるルーティング :

- 隣接 peer に限定
- ObjectID に近い peer に転送



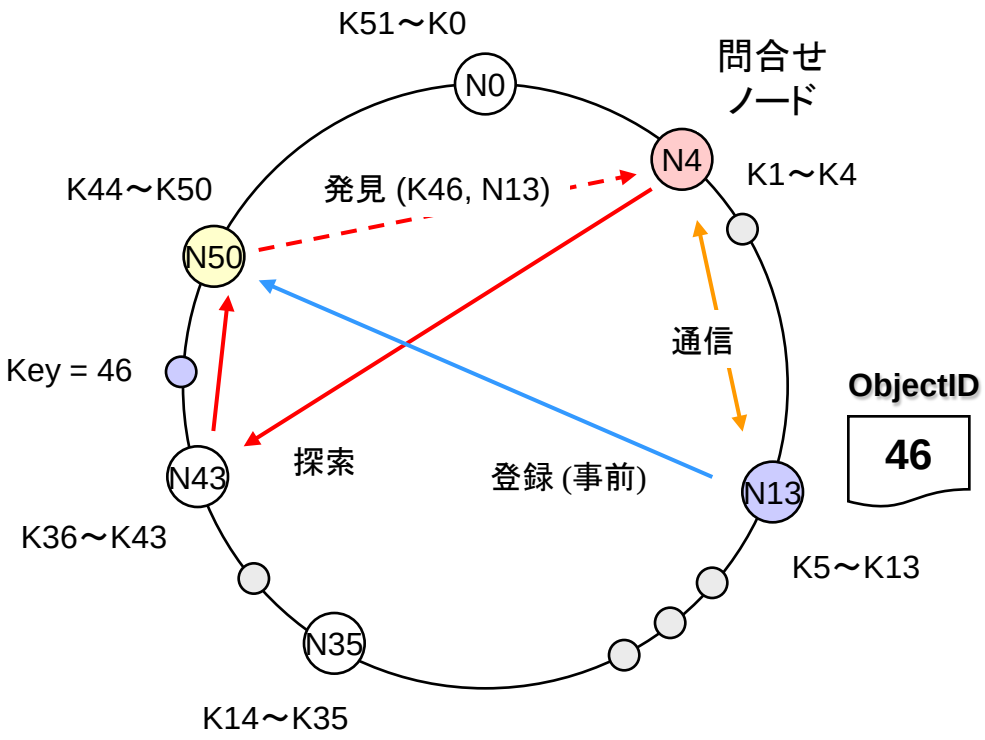
S.Ratnasamy et al: "A Scalable Content-Addressable Network," SIGCOMM'01.

P2P (6) Chord

Plaxton's Algorithm の変形、拡張

各ノードは、1次元円周上の特定の範囲の ObjectID を有するファイルの保有ノード情報を保持

(例) Key (ObjectID) = 46 の探索：
ノード数64、NodeID = 0, 4, 13, 35, 43, 50 の場合



Node 4 のfinger table

Key	Interval	Successor
5 (=4+2^0)	[5,6)	13
6 (=4+2^1)	[6,8)	13
8 (=4+2^2)	[8,12)	13
12 (=4+2^3)	[12,20)	13
20 (=4+2^4)	[20,36)	35
36 (=4+2^5)	[36,4)	43

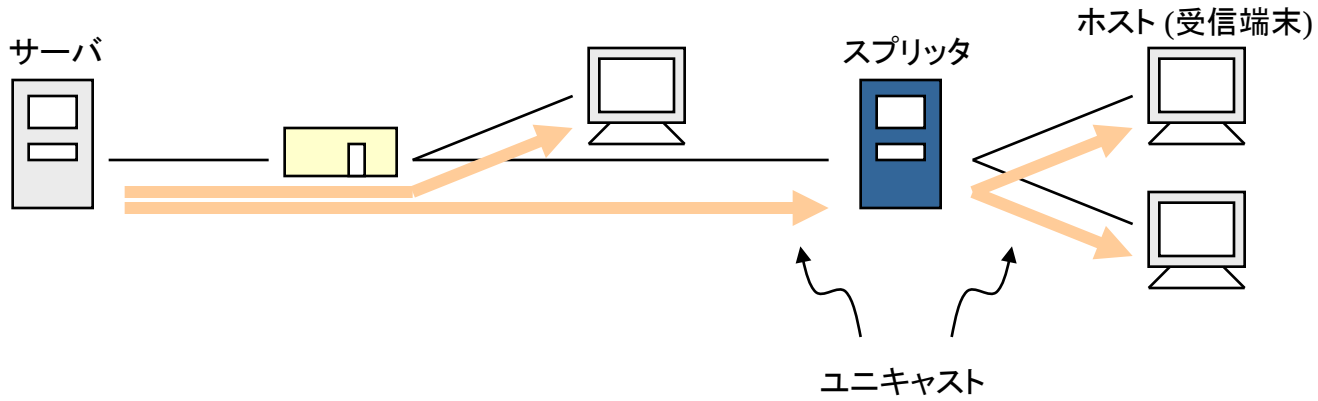
Node 43 のfinger table

Key	Interval	Successor
44 (=43+2^0)	[44,45)	50
46 (=43+2^1)	[46,48)	50
48 (=43+2^2)	[48,51)	50
51 (=43+2^3)	[51,59)	0
59 (=43+2^4)	[59,11)	0
11 (=43+2^5)	[11,43)	13

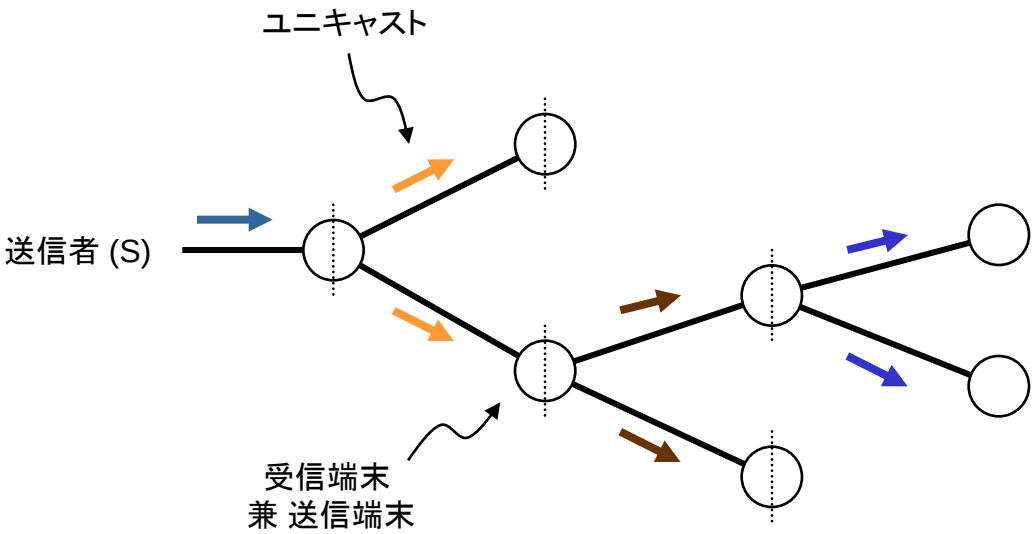
I.Stoica et al: "Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications," SIGCOMM'01.

アプリケーション層マルチキャスト (1)

- スプリッタ

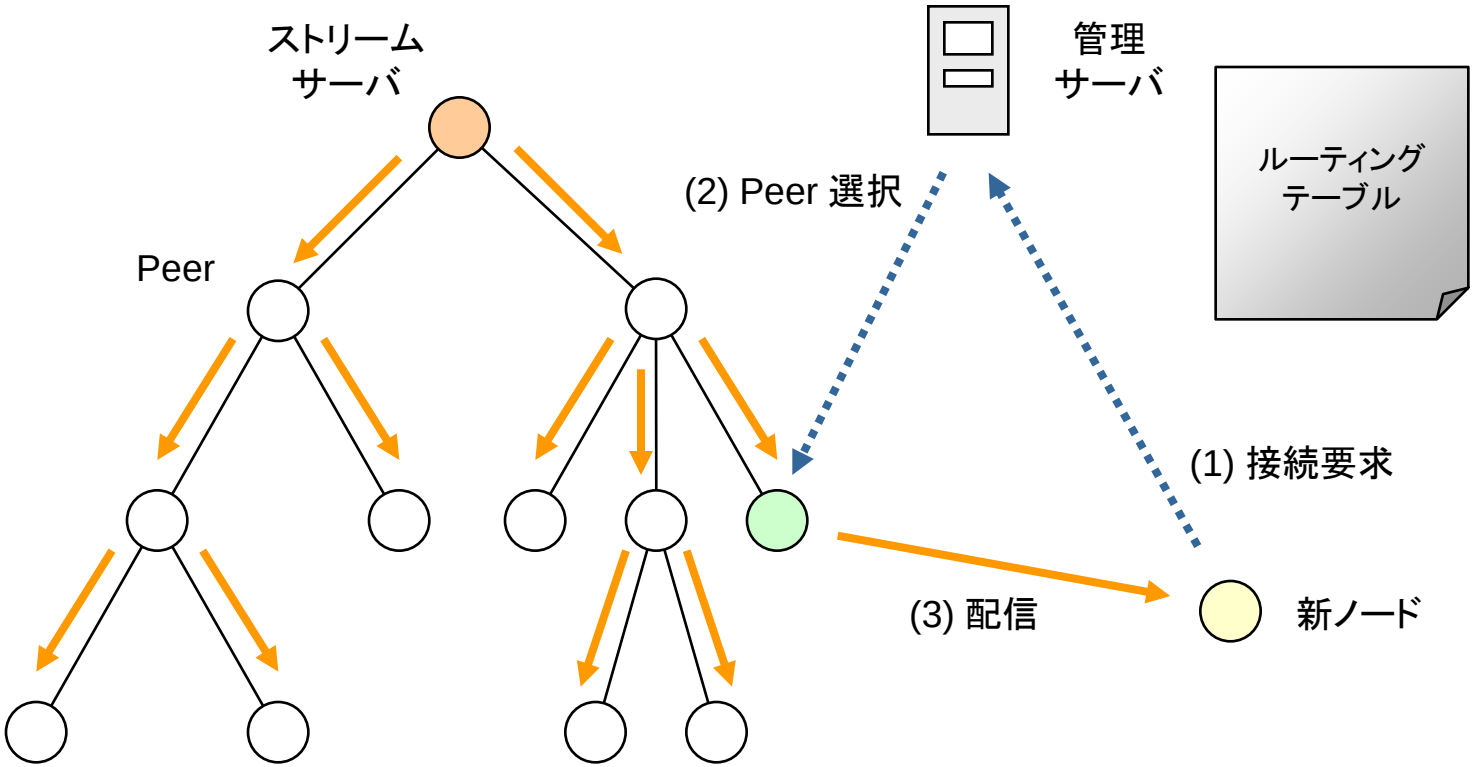


- P2P (Peer-to-Peer)



アプリケーション層マルチキャスト (2)

- P2Pマルチキャスト



長所: 簡単、既存ルータの変更不要

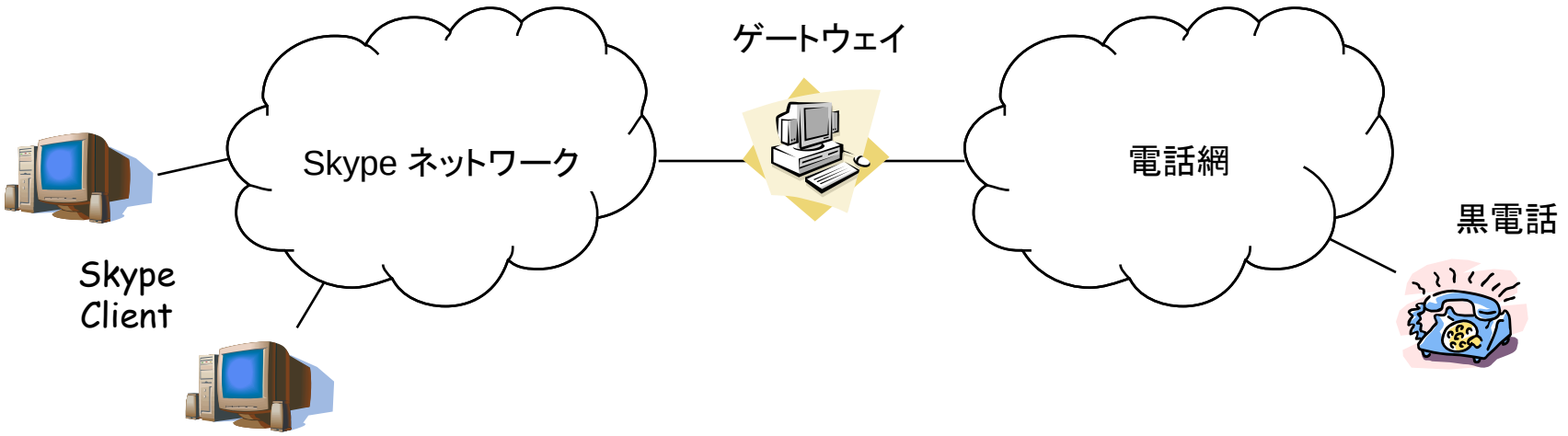
短所: 転送トラヒックの増加、経路の準最適性、管理サーバの負荷

検討事項: ノードの追加と削除への対応、動的な経路変更、負荷分散

Skype (1)

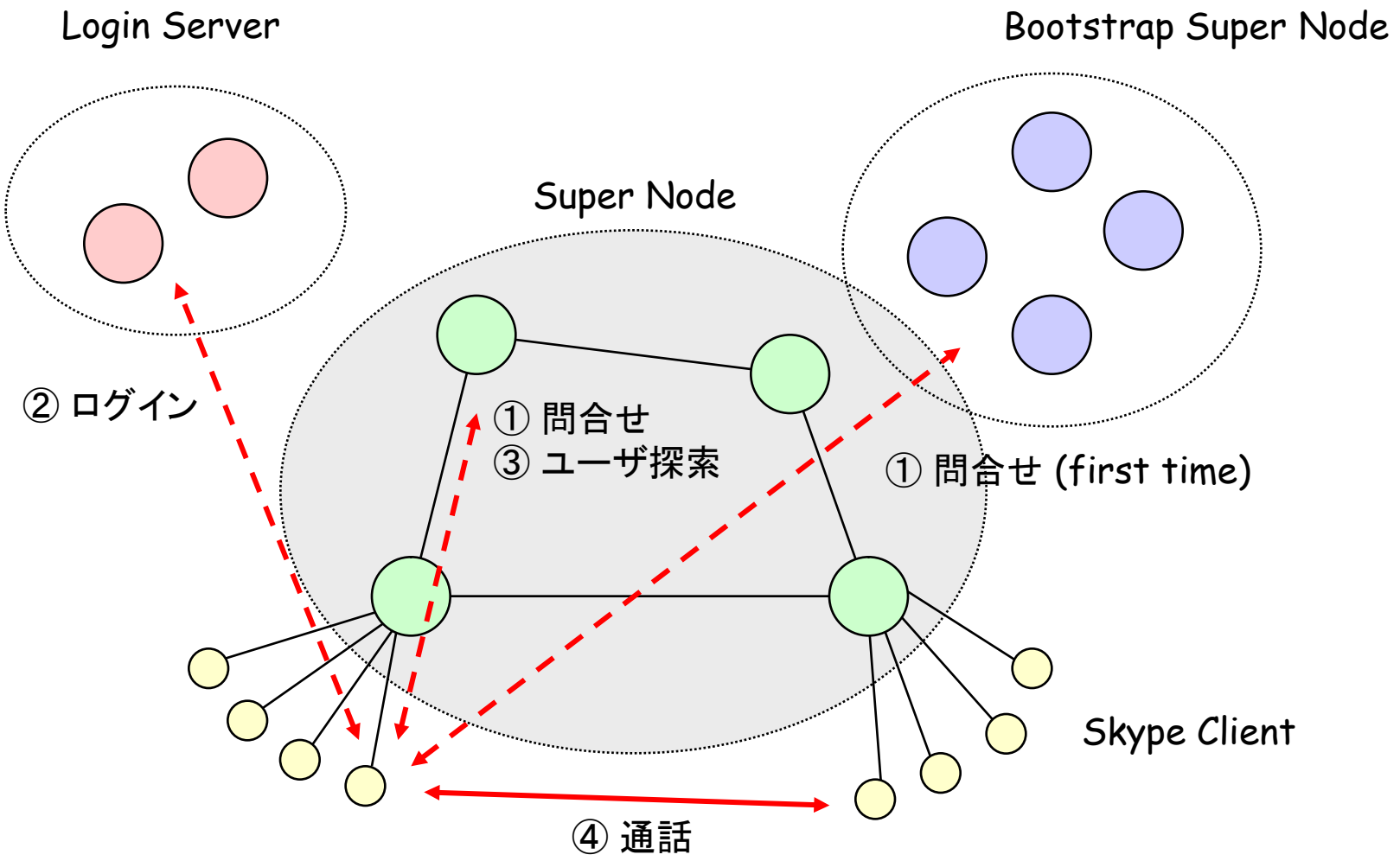
- **P2P 型 VoIP システム**

  - 音質向上 : Global IP Sound (広帯域音声符号化)
  - NAT超え : UDP ⇒ TCP ⇒ HTTP (80) ⇒ HTTPS (443) ⇒ *proxy*
  - 暗号化 : AES (Advanced Encryption Standard, 256 bit)
  - SkypeIn / SkypeOut : 黒電話との発着信



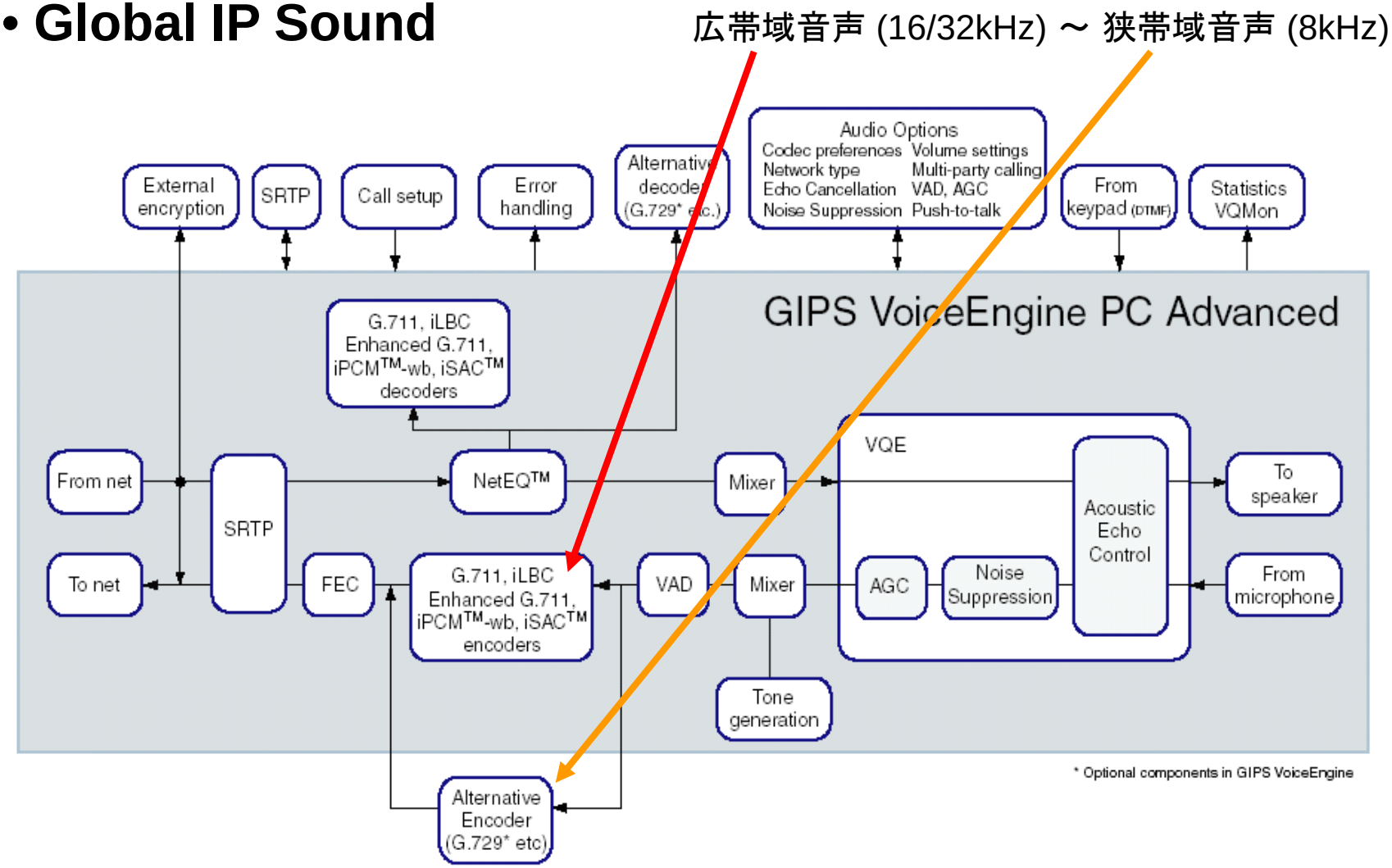
Skype (2)

- システムの構成要素



Skype (3)

- **Global IP Sound**

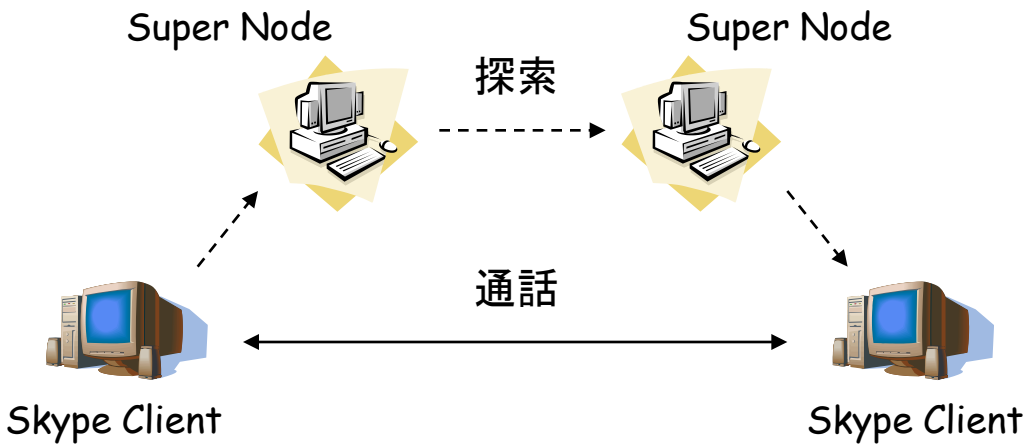


http://www.globalipsound.com/

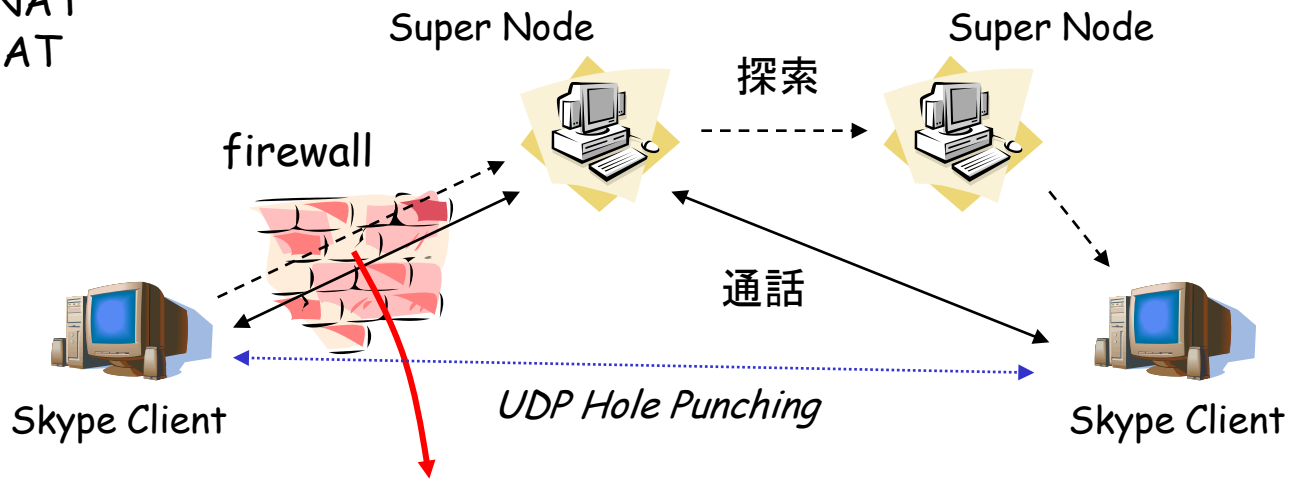
Skype (4)

- **NAT超え**

(1) Public ～ Public



(2) Public ～ NAT
(3) NAT ～ NAT



UDP ⇒ TCP ⇒ HTTP (80) ⇒ HTTPS (443) ⇒ *proxy*