平成 17 年度 修士論文

アドホックネットワークにおけるビデオ配信のための オンデマンド型マルチパスルーティングプロトコル実装

Implementation of an On-demand Multipath Routing Protocol for Video Transmission in Mobile Ad Hoc Networks

早稲田大学大学院 理工学研究科 情報・ネットワーク専攻

甲藤 二郎 研究室

3604U096-1

谷山　健太

受付印	指導教授印

1. Introduction

1.1 Ubiquitous Network

近年，携帯電話や無線 LAN による通信などで、移動中，あるいは外出先でコンピュータを利用する姿が増えてきた．ノートパソコンや携帯端末の高性能化や，携帯電話や PHS によるデータ通信の高速化，無線 LAN アクセスポイントの増加に伴い，モバイルコンピューティングが盛んになってきたのである．こういった携帯端末が今後更なる小型化，高機能化することで，至るところに偏在するネットワーク端末となることも容易に想像されるそのようなネットワーク環境――ユビキタスネットワークにおける技術研究を進めることも非常に重要である．

1.2 MANET

従来の端末の通信の方式では，いったん基地局(BS)や無線 LAN のアクセスポイント(AP)といったインフラストラクチャを介して，他の端末のとの通信をする．しかし、そのような通信では，イベント会場のような人が多く集まる場所や緊急災害時などでインフラが利用できない，また基地局が近くにない，といった一時的な通信が要求される時に役に立たなくなってしまう．このとき，BS/AP とそれを結ぶ通信網によって構成されるような通信でなく，個人の所有するノードが，お互いワイヤレスで直接，同時に複数のノードと通信できるための技術が，アドホックネットワーク(Mobile Ad Hoc Network : MANET)である．MANET はユビキタスネットワーク社会の実現に大きく貢献する技術だと考えられる．

1.2.1 MANET HISTORY

MANET の起源は 1970 年代，インターネットの誕生と同時期まで遡る．1972-1983 年，DARPA（米国国防総省）において，ALOHA プロジェクトや PRNET(Packet Radio Network)の活動における，無線パケットスイッチの技術（帯域共有、パケット中継）の研究プログラムとして始まった．その後，1983-1992 年に SURAN (Survivable Adaptive Networks)における LPR (Low-cost Packet Radio) 技術，1995-2000 年に GLOMO (Global Mobile Information Systems)における無線ブロードキャスト特性を生かした、シングル/マルチホップ通信技術として，軍事利用で研究が行われてきた．

民間に転用されたのは，1990 年代，インターネットの普及と同時期であった．1994 年の ACM SIGCOMM でのルーティングプロトコルの提案**[1]**や CMU Monarch プロジェクトによる研究プラットフォームの提供を受け，1997 年に IETF MANET Working Group**[2]**で，経路制御のプロトコル標準化がスタートする．そして，2003 年には経路制御プロトコルの標準化(Experimental RFC 化)が終了した．

日本国内では，2004 年に，産学官連携のアドホックネットワーク・コンソーシアムの設立や，電子情報通信学会にアドホックネットワーク時限専門研究会の設置が行われるなど研究面で高

まりを見せ始め，また，携帯電話でも NTT DoCoMo や au のプッシュトークサービス(サーバ/クライアント型 擬似アドホック)なども出現し始めている．

1.2.2 MANET Routing Protocols

MANET では，各ノードがルータの役割をすることによってマルチホップの無線通信を行い，エンド間の通信路を確保するが，ノードの移動や不安定な電波特性により，通信路中のリンクブレークが頻繁に起こりやすく，ネットワークトポロジが変化しやすい．このため，IETF MANET WG では，様々なルーティングプロトコルを Experimental RFC 化**[5-8]**してきた．現在，

- RMP (Reactive MANET Protocol)
- PMP (Proactive MANET Protocol)

という 2 つのタイプのルーティングプロトコルが議論され，標準化が進んでいる．RMP は DYMO**[3]**，PMP は OLSR ver.2**[4]**を基に考えられており，これらルーティングプロトコルはいずれも Experimental RFC となった AODV**[5]**,OLSR**[7]**を発展させて考えられたものである．DYMO のベースである AODV は低ルーティングオーバーヘッドやトポロジ変化に強いといった特長を持ち，広く研究されてきた．AODV はリアクティブ型(オンデマンド型)のシングルパスルーティングプロトコルなので，リンクブレークが起きるとルートを再構築する必要がある．

1.2.3 Multipath Routing Protocols

アドホックネットワークでは端末の移動が頻繁に起こるため，データ送信元の端末では経路を再構築しなくてはならない．多くの経路探索することは，経路構築するまでの待ち時間，無線ネットワークで貴重な通信帯域幅とバッテリーを消費してしまうことになる．さらに，ルーティングオーバーヘッドの増加によって，他のデータ通信との干渉が起きてしまい，ネットワーク全体の性能を落としてしまうことになる．このような状況を避けるために，送信元から宛先への経路を複数保持するマルチパス構築型のルーティングプロトコルが提案･研究され，シミュレーション評価されてきた．これらの研究ではデータ送信ルートと重複しない(ディスジョイントな)バックアップルートを保持することで通信の素早い回復を図っている．結果，マルチパスルーティングプロトコルはシングルルーティングプロトコルよりも良い性能を示しているが，実世界でのマルチパスルーティングプロトコルの使用に対する評価としては十分ではないと考える．

また，経路再探索時にはコネクションを張りなおすということは，ストリーミングのようなアプリケーションに大きな影響を与える．リンクブレーク時の回復時間を早めることが，このようなアプリケーションとアドホックネットワークとの親和性を図る方策であると考える．

1.3 Structure of this thesis

以上より，本論文では，オンデマンド型のマルチパスルーティングプロトコルを実装してその有用性を示すと共に，ビデオ配信アプリケーションを使用した際にどのような特性を表すかを評価する．

本論文の構成としては，

第１章は本章であり，MANET の状況及び研究目的を述べている．

第２章では，研究背景及び従来手法について示す．

第３章では，提案手法について述べる．

第４章では，提案手法の実装テストベッドを示すと共に性能評価，並びにビデオ配信デモンストレーションを行う．

第５章では，まとめと今後の展望について述べる．

2. Related Work

2.1 MANET Routing Protocols

MANETでは、各MNがそれぞれ独立に移動したり通信したりするため，ネットワークトポロジが複雑に変化する．また，端末同士の電波干渉や，無線帯域/電力などリソースの有効利用，ルーティングループの回避など，有線では問題にならなかったことまでも考慮する必要がある．したがって，各ノード間のルーティング情報を交換するために，有線で一般的に利用されているRIP(Routing Information Protocol)やOSPF(Open Shortest Path Fast)などによりルーティングプロトコルをそのまま利用すると，大きな負荷がかかる．そこで，このRIPをMANET向けに変更したプロトコルDSDV(Destination Sequenced Distance Vector)**[1]**と，OSPFを変更したOLSR(Optimized Link State Routing)**[7]**，また，差分情報やグラフ理論を積極的に利用したTBRPF(Topology Dissemination Based on Reverse-Path Forwarding)**[6]**といった手法がMANETには存在する．これらの手法は，通信する前にあらかじめ経路を確定しておくといった特徴から，プロアクティブ型ルーティングプロトコルと呼ばれる．

有線では各ルータが固定であることなどから，プロアクティブ型のルーティングプロトコルが使われるが，MANETではルーティング情報交換のために無線帯域を占有することによる影響を抑えるために，通信を行いたい時にのみ経路を探す，リアクティブ型（オンデマンド型）のルーティングプロトコルも考えられている．これにあたるプロトコルとして、AODV(Ad hoc On-Demand Distance Vector)**[5]**とDSR(Dynamic Source Routing)**[6]**などがある。

また，プロアクティブ型とリアクティブ型を組み合わせ部分的に使い分けるハイブリッド型ルーティングプロトコルも考えられ，これにはZRP(Zone Routing Protocol)**[9]**がある．

プロアクティブ型は，あらかじめ経路が確定されているために，データ送信開始時に遅延(待ち時間)が少ないが，通信をしない場合も常に経路情報交換・保持しておく必要があるため，一般的にオーバーヘッドが大きい．リアクティブ型は事前に経路を保持しないために，データ送信時に若干の遅延は避けられないが，経路情報が必要になった時にだけ動作するプロトコルであるのでオーバーヘッドが小さく，帯域や電力消費の面でリアクティブ型に比べて利点があるといえる．ハイブリッド型のZRPはローカルなネットワークでは良いパフォーマンスを示すが，大規模になるとオーバーヘッドが大きくなり，スケーラビリティに欠ける．**表 2.1.1** にこれらの特徴をまとめる．

表 2.1.1 各ルーティングプロトコル特徴

構築タイプ	例	特徴
Reactive (On-demand)	・ AODV ・ DSR	事前データ送信開始時に多少遅延があるが，オーバーヘッドは小さい
Procative	・ DSDV ・ OLSR ・ TBRPF	低遅延だが常に経路情報を交換し続けるため．オーバーヘッドは大きい
Hybrid	・ ZRP	局所的には良いパフォーマンスだが，大規模になるほどオーバーヘッドが大きくなり，スケーラビリティに問題性

2.1.1 AODV (Ad hoc On-demand Distance Vector Routing)[5]

本節では代表的なオンデマンド型ルーティングプロトコルの AODV の動作について述べる.

1) Overview

Ad hoc On-Demand Distance Vector (AODV) ルーティングプロトコルは, アドホックネットワーク内の移動端末のためのルーティングプロトコルである. 動的なリンク状態への素早い適応, 処理とメモリの低負荷, ネットワークの低利用率という特徴を持ち, アドホックネットワーク内における宛先へのユニキャスト経路を決定する. AODV では宛先シーケンス番号を用いることで, 従来の Distance Vector プロトコルで発生する問題(例えば無限カウント)を避けて, 常にループフリーを保証している.

2) Route Discovery Phase

A. Process of Route Request

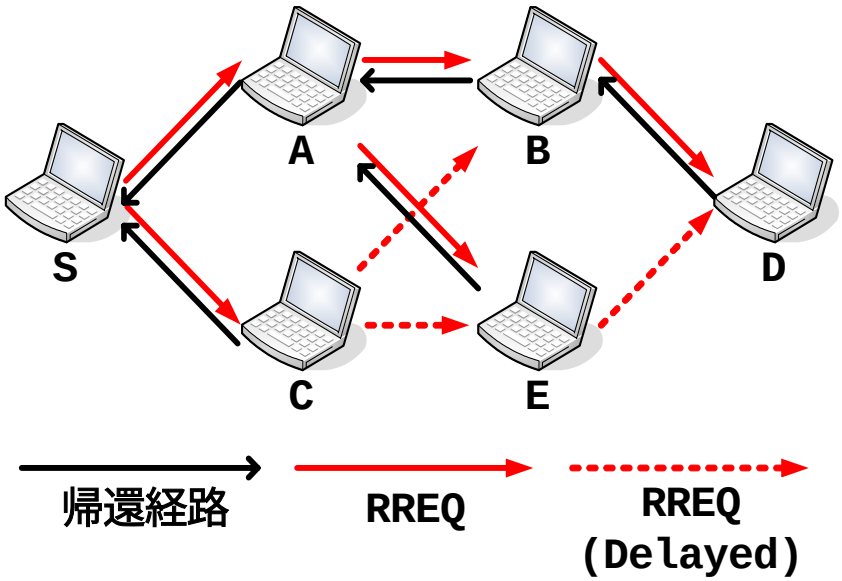


図 2.1.1 RREQ による帰還経路の作成

送信元(Source)S は宛先(Destination)D への経路が必要になった場合, Route Discovery を開始する. S が D へ至るための次ホップのノード(以下, 転送経路)を発見するためにブロードキャストする Route Request メッセージ(以下, RREQ)は, それを受信したノードに対して S へ至るための次ホップのノード(以下, 帰還経路)を与える. **図 2.1.1** のノード A は送信元 S から RREQ を受信することで帰還経路を S, ノード B はノード A から RREQ を受信することで帰還経路をノード A とし, ルーティングテーブルに next hop としてセットする. このとき, データパケットの転送に利用されない経路は, 一定時間後に無効とされ, 後に削除される.

各ノードはシーケンス番号及び RREQ ID と呼ばれる番号をそれぞれ管理する. シーケンス番号は, ループフリーな新しい経路を与えるために用いられ, Route Discovery を行なう S は探索を行なう度に自身のシーケンス番号をインクリメントした値を RREQ に格納する. また, RREQ ID は, 各ノードが以前に受信した RREQ と後に受信した RREQ(以下, Delayed RREQ)が重

複するか否かを判断するために用いられ，シーケンス番号と同様に Route Discovery を行なう度に S は RREQ ID をインクリメントさせ，RREQ に格納する．各ノードは，以前受信した RREQ の RREQ ID と Delayed RREQ に示された RREQ ID が同一の場合，重複する RREQ と判断して，Delayed RREQ を即座に廃棄し，帰還経路の作成及び RREQ の再ブロードキャストを行なわない．例えば，**図 2.1.1** のノード B のようにノード A から RREQ 受信した後にノード C から RREQ を受信した場合，RREQ に示される送信元 S の RREQ ID が同一であることから重複する RREQ メッセージと判断し，廃棄する．

図 2.1.2，**表 2.1.2** に RREQ のメッセージフォーマットを示す．

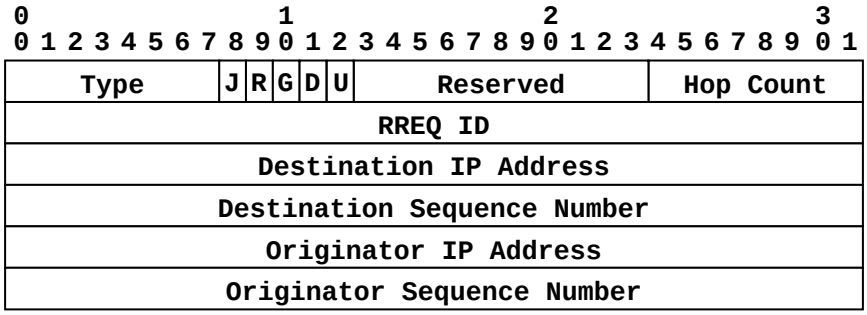


図 2.1.2 RREQ メッセージフォーマット

表 2.1.2 RREQ メッセージのフォーマットフィールド解説

Type	1
J	Join Flag: マルチキャスト用に予約
R	Repair Flag: マルチキャスト用に予約
G	Gratuitous RREP Flag: Destination IP Address フィールド内で指定されるノードへ Gratuitous RREP をユニキャストすべきかどうか
D	Destination Only Flag: 宛先のみがこの RREQ に応答する
U	Unknown Sequence Number: 宛先シーケンス番号が不明
Reserved	0 で送信される．受信時は無視される
Hop Count	生成ノードの IP アドレスから要求を扱うノードまでのホップ数
RREQ ID	生成ノードの IP アドレスと併用して特定の RREQ を識別するような唯一のシーケンス番号
Destination IP Address	ルートの宛先の IP アドレス
Destination Sequence Number	生成ノードがこれまでに受信した宛先方向への(全)ルートのシーケンス番号の中で最新のもの
Originator IP Address	RREQ を生成したノードの IP アドレス
Originator Sequence Number	RREQ を生成したノードを指し示すようなルートエントリ内で使用される現在のシーケンス番号

B. Process of Route Reply

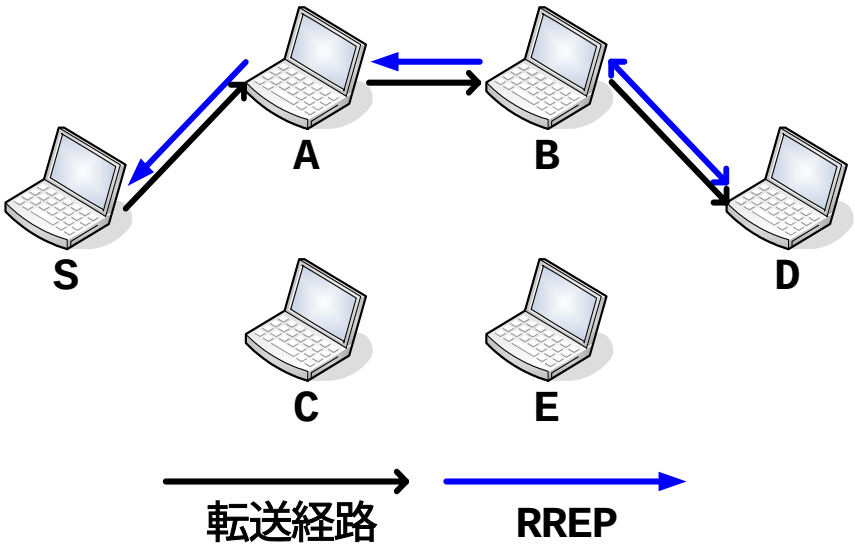


図 2.1.3 RREP による帰還経路の作成

RREQ を受信したノードは次の場合に Route Reply メッセージ(以下, RREP)を生成する.
(i) 自身が RREQ の宛先である
(ii) 宛先へのアクティブな経路を保持していて, ルーティングテーブル内の宛先シーケンス番号が RREQ 内の宛先シーケンス番号以上である, そして, D(宛先ノード限定)フラグがセットされていない

これらどちらかの条件を満たしたノードは, RREQ を再ブロードキャストせず, 作成された帰還経路に沿って送信先 S に向かって RREP をユニキャストする. RREP を受信したノードは, 宛先への転送経路を持っていない場合はルーティングテーブルエントリを作成し, 以下の場合はルーティングテーブルを更新する.
(i) RREP の宛先シーケンス番号がルーティングテーブルの送信先シーケンス番号より大きい
(ii) 宛先シーケンス番号は同じでだが, その経路がアクティブでない
(iii)宛先シーケンス番号が同じであり, なおかつ RREP 内のホップカウントがルーティングテーブル内のホップカウントより小さい

例えば, **図 2.1.3** のノード B は宛先 D から RREP を受信することで転送経路を D とし, RREQ を転送することによって作成された帰還経路に沿ってノード A に RREP を転送する. こうして D から S に至るための帰還経路は 1 つとなり, その帰還経路に沿って RREP が S に至る. 結果, S から D に至る転送経路が決定する. このとき作成された転送経路及び帰還経路はデータパケットを転送する度に, その経路の有効期限を更新する.

図 2.1.4, **表 2.1.3** に RREP のメッセージフォーマットを示す.

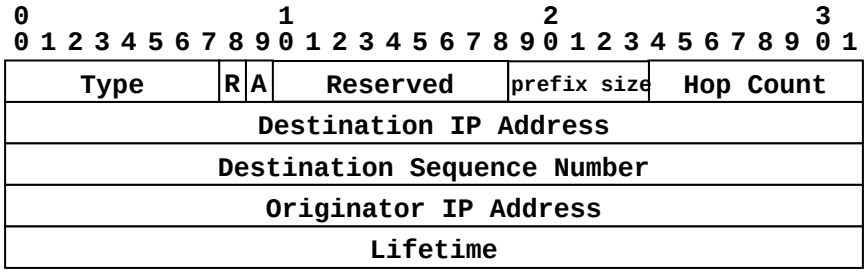


図 2.1.4 RREP メッセージフォーマット

表 2.1.3 RREP メッセージフォーマットフィールド

Type	2
R	Repair Flag: マルチキャスト用に利用される
A	Acknowledgement Required: RREP-ACK メッセージ返送要求
Reserved	0 で送信される. 受信時は無視される
Prefix Size	0 でない時, 5 ビットのプレフィックスサイズは, 次ホップがルータである場合, ルータ以下のサブネットと, 要求された宛先とを同じルーティングプレフィックスにするために使われる
Hop Count	生成ノード IP アドレスから宛先 IP アドレスまでのホップ数のこと. マルチキャストルートリクエストの場合, これは RREP を送信したマルチキャストツリーメンバーまでのホップ数を示している
Destination IP Address	ルートが与えられる宛先 IP アドレス
Destination Sequence Number	ルートに関連した宛先シーケンス番号
Originator IP Address	そのノードのためにルートが与えられる, RREQ を生成したノードの IP アドレスのこと
Lifetime	RREP を受信したノードが, そのルートが有効であると考えるミリ時間

2) Route Maintenance Phase

A. Process of Route Error

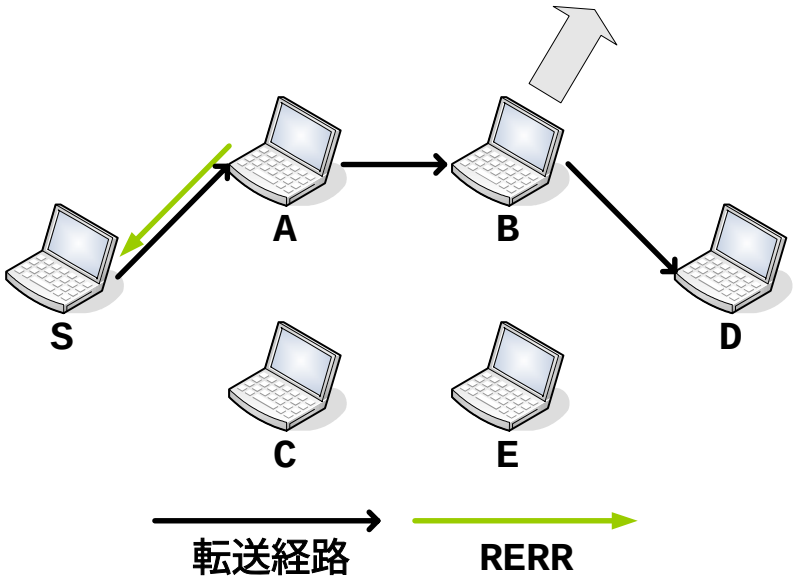


図 2.1.5 RERR による経路削除

宛先や中間ノードが移動した場合，Route Error メッセージ(以下，RERR)が影響するデータの送信元に向かって送信される．以下の場合において，RERR の処理を開始する。

(i) ルーティングテーブルにおけるアクティブルート上の次ホップノードへのリンクブレークを検出した

(ii) 宛先へのアクティブなルートを持たず，(Local Repair プロセスを利用しても) 修復できなかったノード宛のデータパケットを受信した

(iii) 1 つ以上のアクティブルート上の隣接ノードから RERR を受信した

いずれの場合においても，もしもこのノードが 1 つ以上の宛先へのプリカーソルノードを持っているならば，非到達宛先ノードのアドレスを含んだ RERR を近隣ノードにブロードキャストする．プリカーソルノードとは，RREP が転送されてくる近隣ノードのことであり，リンク切断を検知した場合に，検知したノードからの知らせを受信するノードである．RERR を受信したノードは，このメッセージの中に含まれている非到達宛先ノードに対する経路表内の経路を無効にする．そして順番に RERR をプリカーソルノードに伝達させていく．送信元が RERR を受信したとき，続けて通信を行なうために経路が必要な場合は，Route Discovery フェーズを再び開始する．

図 2.1.5 で RERR プロセスの様子を示した。**図 2.1.5** において，送信元 S から宛先 D への経路はノード A,B を通っている．ノード B が移動すると，ノード A との間でリンク切断が起きる．ノード A はリンク切断を検知すると，D へのプリカーソルノードであるノード S に RERR を送信する．RERR を受信した S は経路が必要であるので，経路を再探索し，**図 2.1.5** においてノード C を通った新しい経路を作成する．

図 2.1.6, **表 2.1.4** に RERR メッセージフォーマットを示す．

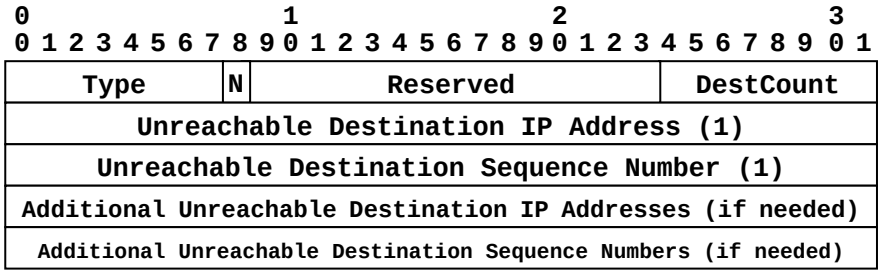


図 2.1.6 RERR メッセージフォーマット

表 2.1.4 RERR メッセージフォーマットフィールド

Type	3
N	No Delete Flag:ローカルリペアを実行している時に設定される. 上流(S 側)ノードはルートを削除すべきではない
Reserved	0 で送信される. 受信時は無視される
DestCount	メッセージ内に含まれる非到達宛先の数のこと. 少なくとも 1 以上
Unreachable Destination IP Address	リンクブレークのために到達性がなくなった宛先の IP アドレス
Unreachable Destination Sequence Number	以前の Unreachable Destination IP Address フィールド内に記載された宛先のためのルートテーブルエントリ内のシーケンス番号

4) AODV Option

-Local Repair オプション

アクティブルートでリンクブレークが発生した時, その上流(S 側)ノードは宛先まで MAX_REPAIR_TTL ホップ以内であれば, 局所的に経路を修復することができる. このとき, その上流ノードはあて先に対して RREQ をブロードキャストする. 修復を開始したノードは, RREQ に応答した RREP を受信するためにある一定期間待つことが必要である. Local Repair の間, 送信されてきたパケットはキャッシュされる. もしある一定期間が過ぎても RREP を受信しない場合, 3)のように RERR を送信する.

また, ある一定期間に修復を開始したノードが 1 つ以上の RREP を受信した場合, 宛先に対するルーティングテーブルのホップカウントと RREP 内の新しいホップカウントを比較する. もしも宛先への新しく決定された経路のホップカウントが, 以前作成された経路のホップカウントより大きければ, そのノードはその宛先に対する RERR を生成するべきである. それ以外の場合は, ルーティングテーブルの経路を更新して, 2)で述べられた様に処理をする. これにより局所的に経路を修復することができ, Local Repair 中にキャッシュされたパケットを送信することができる.

図 2.1.7 は B が移動してしまった際に, ノード G において Local Repair が行われる様子を示している.

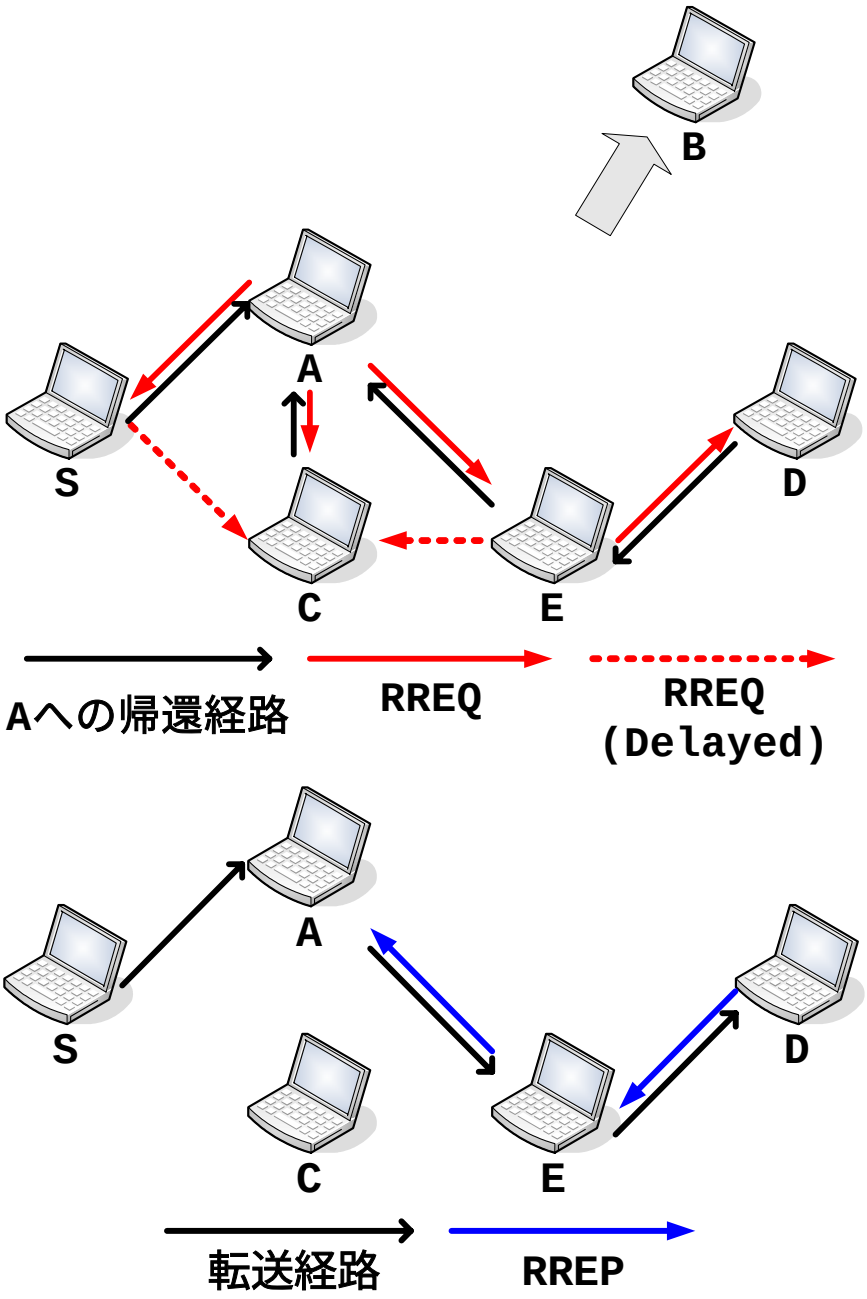


図 2.1.7 Local Repair

-Hello オプション

それぞれのノードは局所的に Hello メッセージ(以下, Hello)をブロードキャストすることで接続性情報を確認しても良いが，それはアクティブルートに対して行うのが良い．HELLO_INTERVAL 以降，HELLO_INTERVAL ミリ秒間に，RREQ やリンクレイヤーメッセージなどのブロードキャストパケットを送信したかどうか確認し，送信していなければ Hello(TTL=1 のブロードキャスト版 RREP)を送信してもよい．Hello を受信したノードは，その隣接ノードへの経路がアクティブと考え，ルートエントリを作成または更新する．更新する場合，経路のライフタイムを ALLOWED_HELLO_LOSS * HELLO_INTERVAL 以上に更新する．Hello パケットによっ

て作成されたルートエントリでは，プリカーソルリストは空であり，隣接ノードが検知できなくなったとしても，RERR を送信しない．以下，**表 2.1.5** は RREP パケットを Hello メッセージとして使用した時のフォーマットである．

表 2.1.5　Hello メッセージフォーマット

Hop Count	0 をセット
Destination IP Address	そのノードの IP アドレスをセット
Destination Sequence Number	そのノードの最新のシーケンス番号をセット
Lifetime	ALLOWED_HELLO_LOSS * HELLO_INTERVAL

-Ring Search オプション

不必要な RREQ の拡散を抑えるために、Ring Search オプションが適用することができる．最初にブロードキャストされる RREQ の IP ヘッダの TTL 値を TTL_START(推奨値 5)にセットし，TTL 値から計算された期間(RING_TRAVERSAL_TIME ミリ秒)待っても RREP が返信されてこなければ，TTL 値を TTL_INCREMENT(推奨値 2)だけ増やしていく．TTL 値が TTL_THRESHOLD(推奨値 7)を超えた場合は，TTL 値を以降 NET_DIAMETER(推奨値 35)とする．待ち時間は、TTL 値が増えるごとに増加していく．

無効となったルーティングテーブル内のホップカウントは，その宛先への最新なホップ数であるので，もしもその宛先への経路を再び探索する場合には，RREQ の IP ヘッダの TTL 値は，最初その値+TTL_INCREMENT とする．後は上記と同様な動作を行う．

2.1.2 DSR(The Dynamic Source Routing Protocol)**[6]**

本節では代表的なオンデマンド型ルーティングプロトコルの DSR の動作について述べる．

1) Overview

The Dynamic Source Routing protocol(DSR)はマルチホップワイヤレスアドホックネットワークにおいて機能するように作られた，簡明かつ効果的なルーティングプロトコルである．DSR ではネットワークのインフラや管理をすることなしに，全て自動的にネットワークを構築･設定することができる．プロトコルは全てオンデマンドに動作し，現在使っているルートを変える必要がある場合にのみ働くので，DSR のルーティングパケットによる負荷を抑えることができる．また，宛先への多数のルートを持っており，送信元はそのルートを選択･制御することができる．

2) Route Discovery Phase

A. Process of Route Request

送信元 S が宛先 D への通信を始めたいときに，送信元 S はデータパケットのヘッダに，宛先への転送経路上のノードのアドレス列である「ソースルート」情報をのせる．そして、送信元 S は宛先 D へのルートを自分の「ルートキャッシュ」内に持っているかどうか判別する．もし宛先 D へのルートをもっているならば，そのルートを使ってデータパケットを送る．もしなければ Route Discovery によって直ちに新しいルートの探索がなされる．

まず，ルートの探索のために、送信元 S は RREQ をブロードキャストし，S の通信範囲内にある

ノードは全てこのメッセージを受け取る. この RREQ は宛先 D のアドレス, 送信元 S のアドレス, ID ナンバーを含んでいる. また RREQ は自分が転送されてきた中間ノードのアドレスリスト(ソースルートリスト)も含んでいる.

この RREQ が中間ノード(例えば**図 2.1.8** のノード B)に到達した場合, そのノードは RREQ の ID と宛先のアドレスを自身の「リクエストテーブル」に記録し, また, 自分が宛先 D へのルートを持っていないかルートキャッシュをチェックする. もし持っていなければそのノードは自分のアドレスを RREQ に記録し, RREQ を転送(再ブロードキャスト)する. このとき, RREQ の転送数を制限するためとルートのループ回避のため, 各ノードは,

- リクエストテーブルと対比し、RREQ 内の ID と宛先アドレスが同一の RREQ を以前に受け取っていない もしくは
- RREQ 内に自分のアドレスが含まれていない

場合にのみ, RREQ を転送する. **図 2.1.8** の場合, ノード B において S-A-と RREQ が転送されてきたとする. その後 S-C-という RREQ を受信した場合, ノード B では, ID と宛先アドレスが同一の RREQ を受信したということで S-A-の RREQ を廃棄する。

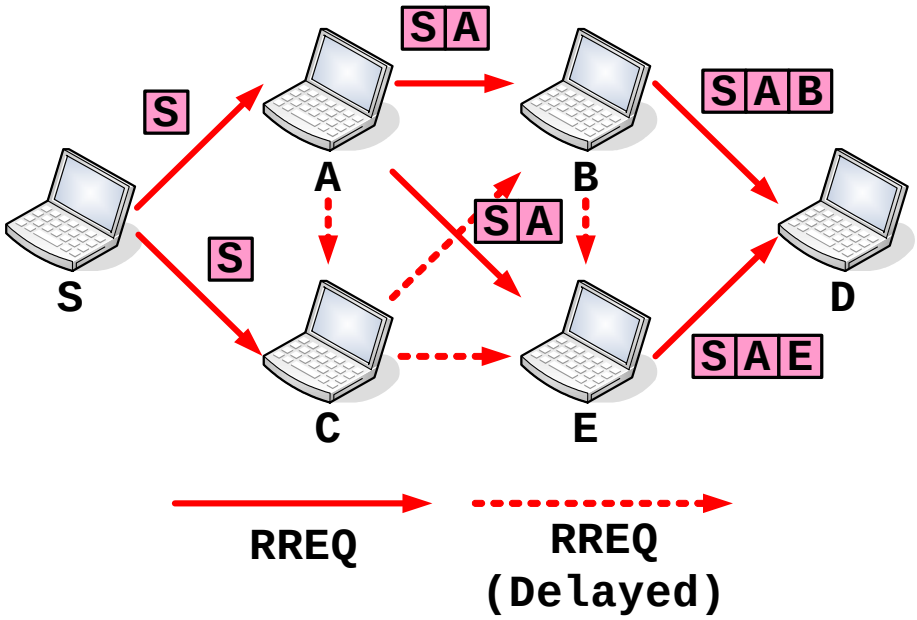


図 2.1.8 RREQ プロセス

B. Process of Route Reply

RREQ の到着時に, そのノードが宛先 D への経路をルートキャッシュ内に保持していた場合もしくは, そのノード自身が宛先 D であった場合に RREP が返される. このノードが宛先 D であった場合, RREQ 内にあるアドレスリストを RREP にコピーして, 送信元 S まで転送する. このノードが中間ノードであった場合には, RREP には RREQ 内にあるアドレスリストに, 自身のルートキャッシュ内にあるルートのアドレスリストを加えて RREP を生成し, 送信元 S まで転送する. **図 2.1.9** では宛先 D が 2 つの RREQ を受信したため, 2 経路に対して RREP が返されている. 宛先ノードでは RREQ を複数受信した場合, 複数の RREP を返信する.

また RREP を転送するノードは, ルートキャッシュに RREP のルートをキャッシュする.

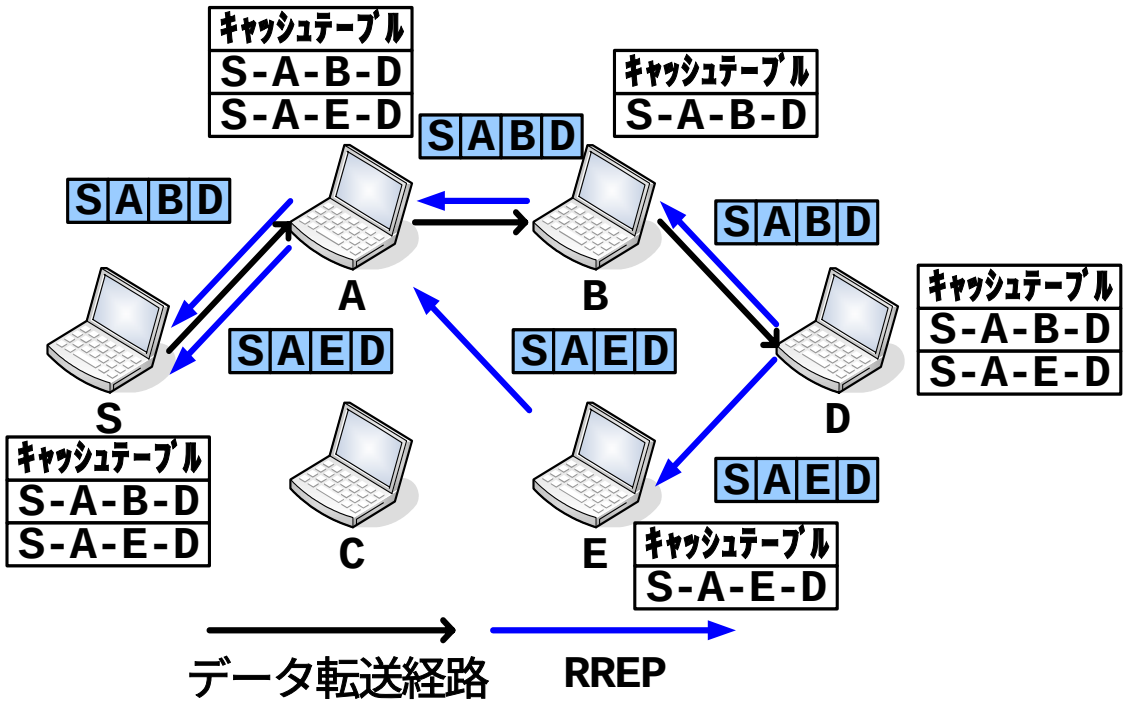


図 2.1.9 RREP

2) Route Maintenance Phase

A. Process of Route Error

ソースルートを使ってパケットの転送を行なうとき，次ホップへの経路が使えるかどうか確認しなければならない．

経路が使えるかの確認は

1.リンク層 Acknowledgement を用いる方法

2.passive acknowledement を用いる方法

3.Acknowledgement Request オプションを用いる方法

の 3 つがある．

IEEE 802.11 のような無線 LAN を使っている場合には，1 のリンク層 ack が使用される事が望ましく，そうでない場合は 2 の方法が利用される．2 の passive acknowledgement とは，次のノードがさらに次のノードへ送信したパケットを盗み聞きする方法のことである．1，2 ともに利用できない場合は，DSR の Acknowledge Request オプションにより確認応答がなされる．

ack が届かなかった場合，パケットを送信したノードは次ホップへのリンクがブレークしたとし，そのリンクをルートキャッシュから削除するとともに RERR をそのリンクを使ってパケットを送っていた各ノードへ送信する．**図 2.1.10** で S-A-B-D で通信をしていて，もし B が移動し，A からの ack を受信できなかった場合，A は S へ RERR を返し，RERR を受信したノードはこのブレークしたリンクをキャッシュ内から削除する．宛先 D への再送や，次のパケットの送信には，もし送信元 S がそのルートキャッシュに別の有効なルート(例えば前に行なわれた Route Discovery による RREP によるものや，他のパケットの盗み聞きによって得られたもの)があれば，即座にその新しいルートを使って通信が始まる(**図 2.1.11**)．もし別ルートがなければ新しく Route

Discovery が始まる.

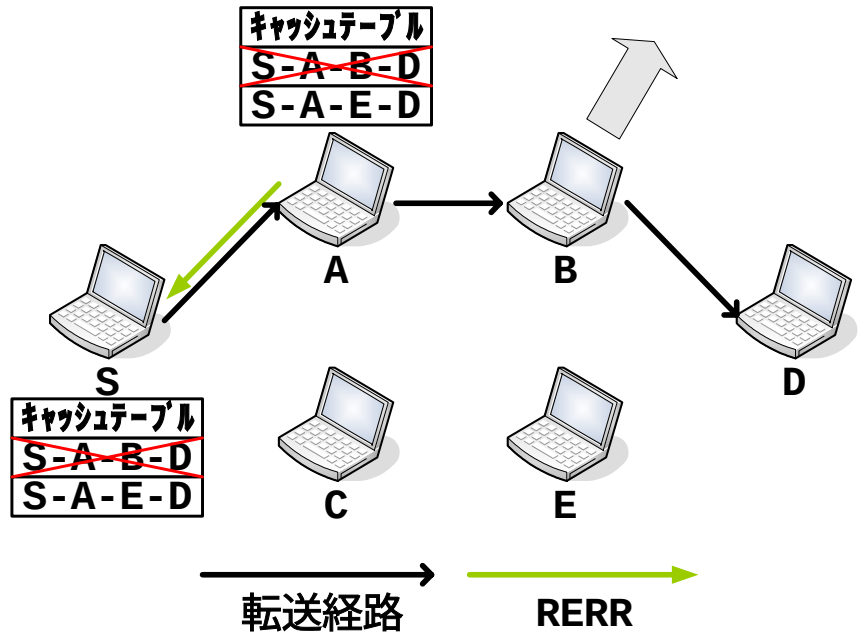


図 2.1.10 リンクブレークによる RERR

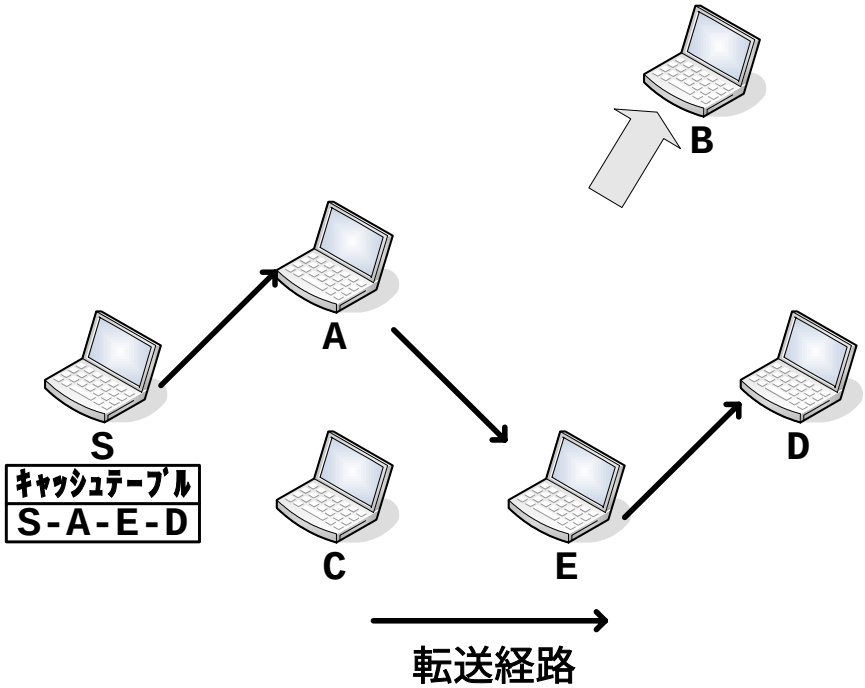


図 2.1.11 RERR 後の復帰

3) Flow State Option

このフローステートでは，ソースルートをパケットヘッダーに含まずに通信を可能にする．まず，Route Discovery によって宛先 D へのソースルートを見つけると，送信元 S はソースルートヘッダを含むパケットを通常の DSR と同じように送る．パケットは D へソースルートに沿って転送さ

れるが，この際，中継ノードはフローステート機能を有効にするため，パケットの転送先のノードのアドレスを記憶してゆく．これにより，後続のパケットは，パケット上にソースルートヘッダを含めることなく，宛先へパケットを転送することが出来るようになる．

フローテーブルには宛先へのフロー毎に {source address, flow identifier} が関連付けて記憶されている．フローテーブルエントリを作成するために，フローステートを開始したいノードはパケット上にソースルートと共に flow identifier を含めて転送する．このようなパケットを受信したノードは新たなフローの情報をフローテーブルに書き込んでゆく．

フローテーブルに情報をもつノードがパケットを送信・転送するときには，ソースルートの代わりにフローヘッダを加える．各ノードがフローヘッダをもつパケットを受信した場合，flow identifier をもとにフローテーブルエントリに記載される次ホップにパケットを転送する．もし，フローテーブルエントリがなければ，RERR を送信元 S にむかって返し，フローテーブルに関連付けられているソースルートとそのルートキャッシュを削除してゆく．

2.2 Multipath Routing Protocols

本節では従来のマルチパスルーティングプロトコル研究について述べる.

2.2.1 Terminology

1) ディスジョイントパス

マルチパスルーティングでは経路構築時に, 重複しないパスを作成することによってリンク切断が独立となり, 耐性が高まる. この重複しないパスのことを“ディスジョイントパス”と呼ぶ. ディスジョイントパスには次に述べる 2 種類がある.

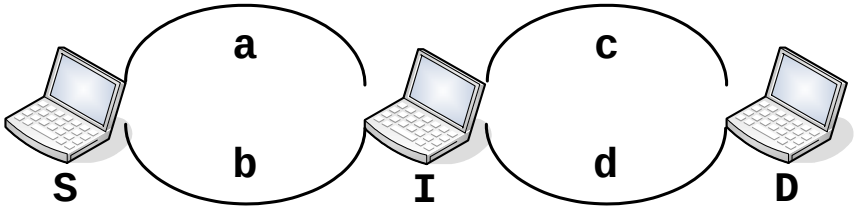


図 2.2.1 ディスジョイントパス

- リンクディスジョイント:**図 2.2.1** において, 経路 a-c と b-d, あるいは経路 a-d と b-c のような, リンクを共有することがないがノードでの共有はある経路
- ノードディスジョイント:**図 2.2.1** において、経路 a と b, あるいは経路 c と d のように, end-end ノード間で全く共有することがない経路

論文[11]では演繹的にリンクディスジョイントパスの方が, ノードディスジョイントパスよりも作成できる可能性が高いことを示しているが, ノードディスジョイントパスは両経路が完全に独立なため, 各々に別のデータを流すという利用法も考えられている.

2) Delayed RREQ/RREP & Bicast

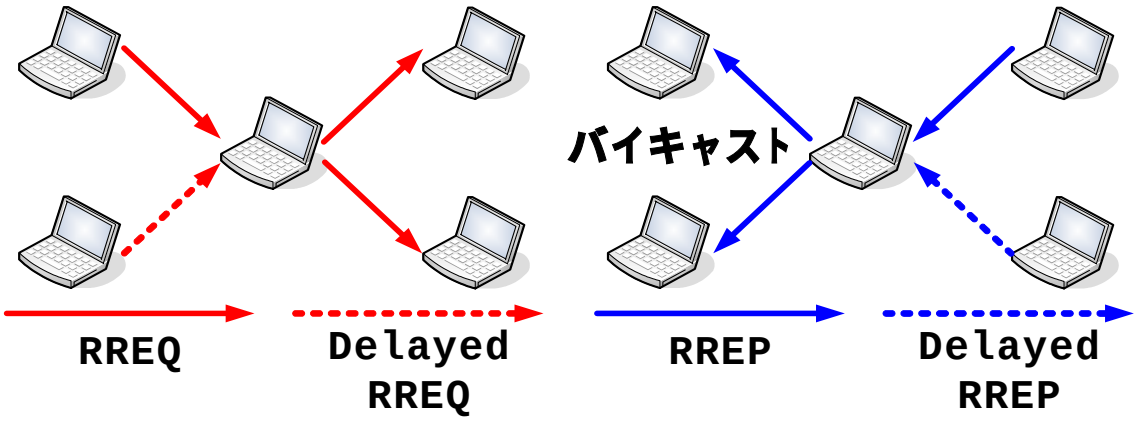


図 2.2.2 バイキャスト

前節 **2.1** で, 遅れて受信された同一 ID を持つ RREQ を Delayed RREQ と呼んでいたが, 本節では遅れて受信された同一シーケンス番号の RREP に関しても, ”Delayed RREP”と呼ぶ. また, マルチパスルーティングの特徴として, 経路を確定するために複数方向に RREP を投げることがあるが, このとき, 受信された RREP ひとつに対し,

・ 一度に一方向に投げることをユニキャスト
・ 一度に複数方向に投げることをバイキャスト**[10]**（もともとは「2 方向にユニキャスト」の意だが，本論文では「複数ユニキャスト」と広義に考える）

と呼ぶことにする（**図 2.2.2**）．

2.2.2 AOMDV(Ad hoc On-demand Multipath Distance Vector)**[11]**

1) Overview

AOMDV は AODV をマルチパスルーティングに拡張したルーティングプロトコルである．**図 2.2.3** は，ルーティングテーブルの拡張を示している．AOMDV では経路のループを防ぐために，advertised_hopcount フィールドを作成する．advertised_hopcount フィールドは，宛先に対する複数の経路のうちの”最大”ホップ数を表したものである．それぞれノードは RREQ/RREP によって転送されてきたホップカウントと advertised_hopcount を比較し，ルーティングエントリ内の advertised_hopcount より小さい値をもった経路のみをバックアップ経路として受け入れる．

また，AOMDV はリンクディスジョイントパスを構築するために，RREQ/RREP に first_hop フィールドを追加する．first_hop フィールドには送信元の隣接ノードのアドレスを挿入する．また，first_hop リストを保持し，RREQ がどの first hop を通過してきたかを記憶する．

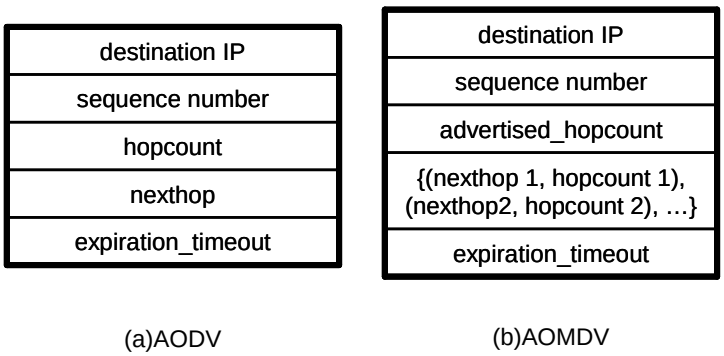


図 2.2.3 AODV と AOMDV のルーティングテーブルエントリ

2) Route Discovery Phase

A. Process of Route Request

図 2.2.4 に RREQ プロセスを示す．送信元 S は RREQ をブロードキャストする．S から RREQ を受信した隣接ノード(図ではノード A,B)は RREQ の first_hop フィールドに自身のアドレスを挿入してこのパケットを転送する．ノード I はノード A から受信した RREQ により帰還経路を作成した後，ノード B から Delayed RREQ を受信した場合，RREQ の first_hop フィールドとノード I が保持する firsthop_list 内のアドレスが異なるため，ノード B への帰還経路を作成する．この際，Delayed RREQ は再ブロードキャストせずに破棄する．また，宛先 D がノード X から受信した RREQ により帰還経路を作成した後，ノード Y から Delayed RREQ を受信した場合，first hop は同一であるが，”looser” reply policy によって，異なるノードから送信さ

れてきた RREQ に対しては，ノード Y への帰還経路を作成することができる．これは k 回まで許され，演繹的に k=3 としている．

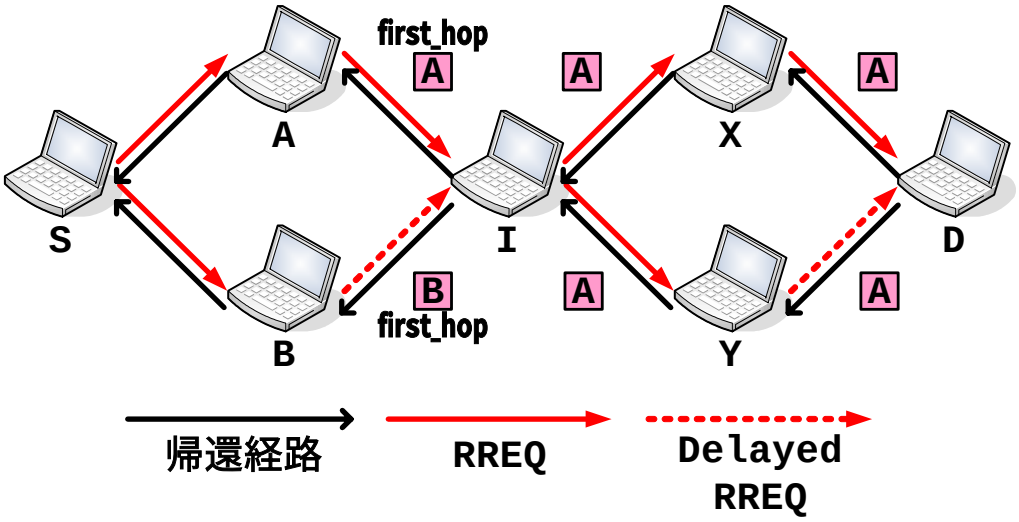


図 2.2.4 RREQ プロセス

B. Process of Route Reply

図 2.2.5 に RREP プロセスを示す．RREQ を受信した宛先 D は，それぞれの RREQ に対して RREP を送信元 S へ送信する．このとき，RREP の first_hop フィールドには RREQ が転送されてきたノードのアドレスを挿入する．ノード I は，ノード X から転送されてきた RREP に対する転送経路を作成し，RREP を複数帰還経路のうちの 1 つへ向けて転送する．その後，ノード I にノード Y から RREP が転送されてきた際には，RREP の first_hop フィールドとノード I の first_hop リストのアドレスが異なるので，ノード Y への転送経路を作成し，別の帰還経路へ RREP を転送する．以上，RREQ/RREP プロセスによって複数の経路が作成されることになる．

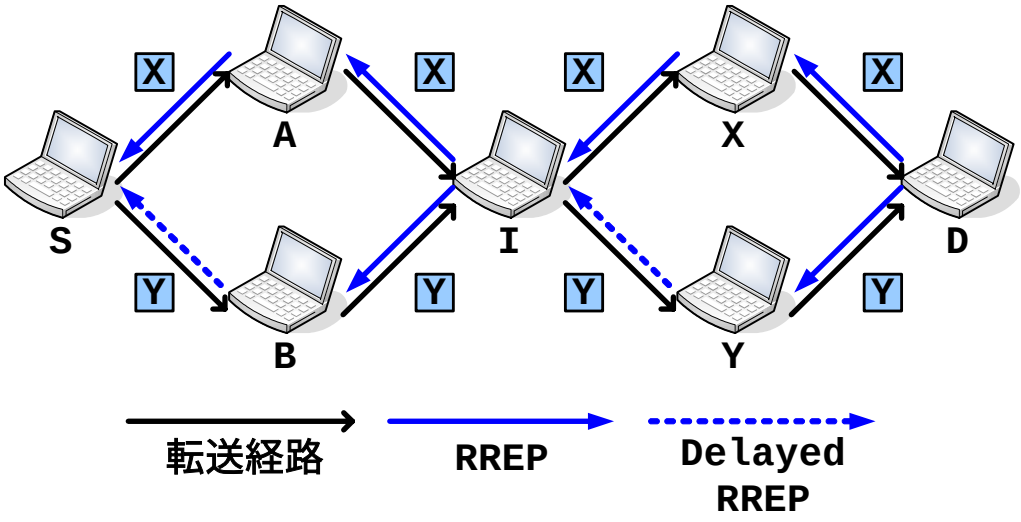


図 2.2.5 RREP プロセス

2) AOMDV の問題点

AOMDV のように送信元 S 及び宛先 D の近隣ノードの情報を用いるだけでは**図 2.2.6** のノード I, J のような中継ノードが 2 つ以上共通する場合に，リンクディスジョイントパスを構築できなくなるという問題がある．ノード I ではノード A から受信した RREQ を再ブロードキャストすると、ノード B から後から受信した RREQ を再ブロードキャストしない．これによって、ノード J はノード C, E から RREQ を受信した場合，first_hop フィールドが異ならないので，1 つの帰還経路しか作成することができない．その結果，図ではノード E とノード J 間の帰還経路が作成されず，S-J 間にディスジョイントパスは構築されないことになる．

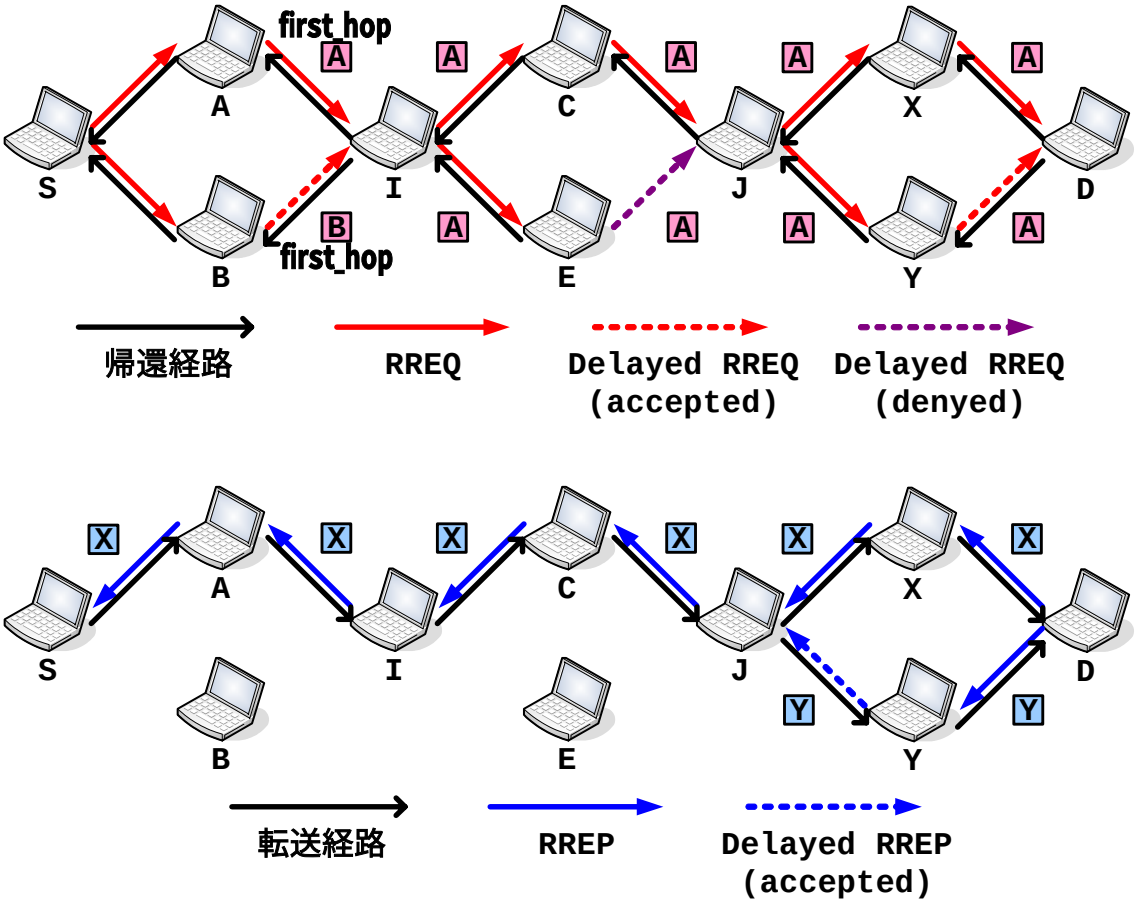


図 2.2.6 AOMDV の問題点 1(上図：RREQ/下図：RREP プロセス)

また，**図 2.2.7** のように 3 ホップ程度の通信では，リンクディスジョイントを作成することが出来ないため，ノードディスジョイントな経路を選択したい．ところが，中間ノードでは RREP を 1 方向にユニキャストすることしか出来ない．よって，複数の帰還経路を RREQ で作成しても，RREP は最初に届いた RREQ に作成された帰還経路しか返すことができない．

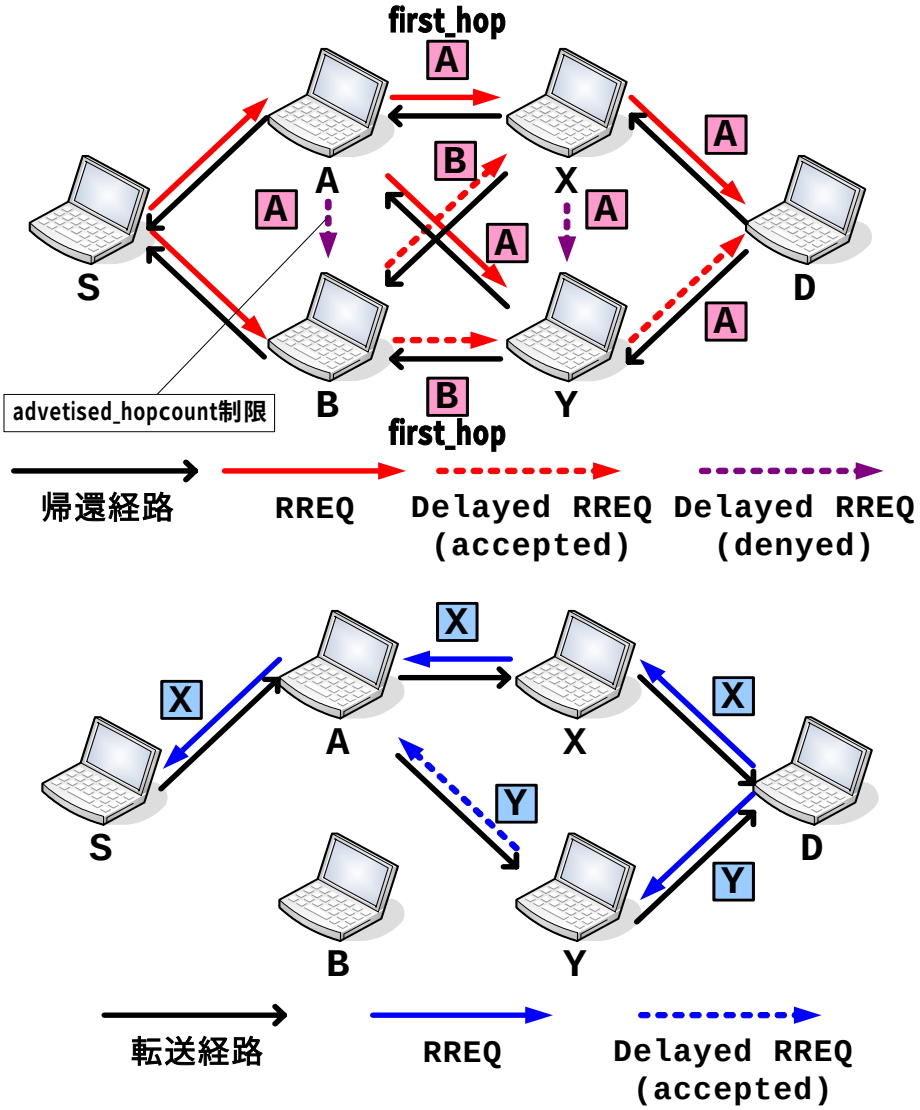


図 2.2.7 AOMDV の問題点 2(上図:RREQ/下図:RREP プロセス)

2.2.3 AODVM(AODV-based Multipath Routing Protocol)[12]

1) Overview

AODV を拡張したマルチパスルーティングである. 本論文では, この提案を AODVM と呼ぶことにする.

AODVM では, 経路のループを回避するため, ルーティングテーブル内に保持されている経路の中で最も大きいホップ数をもつ経路より受信した RREQ/RREP のホップ数が小さい場合のみをバックアップ経路として受け入れる. この点は AOMDV の advertised_hopcount の概念と同一である(本項では advertised_hopcount と呼ぶ).

また, リンクディスジョイントパスを構築するために, AODVM では jointcount と呼ばれる追加フィールドをルーティングテーブルと RREP に追加する.

2) Route Discovery Phase

A. Process of Route Request

送信元 S は RREQ をブロードキャストする. 上述したように advertised_hopcount の制限により, 帰還経路を作成していく. **図 2. 2.8** において, ノード G ではノード A から受信した RREQ によって帰還経路が作成された後, ノード I から Delayed RREQ を受信すると, ノード I への帰還経路を作成する. このとき Delayed RREQ は再ブロードキャストしない. この動作を繰り返して宛先 D まで RREQ を転送する.

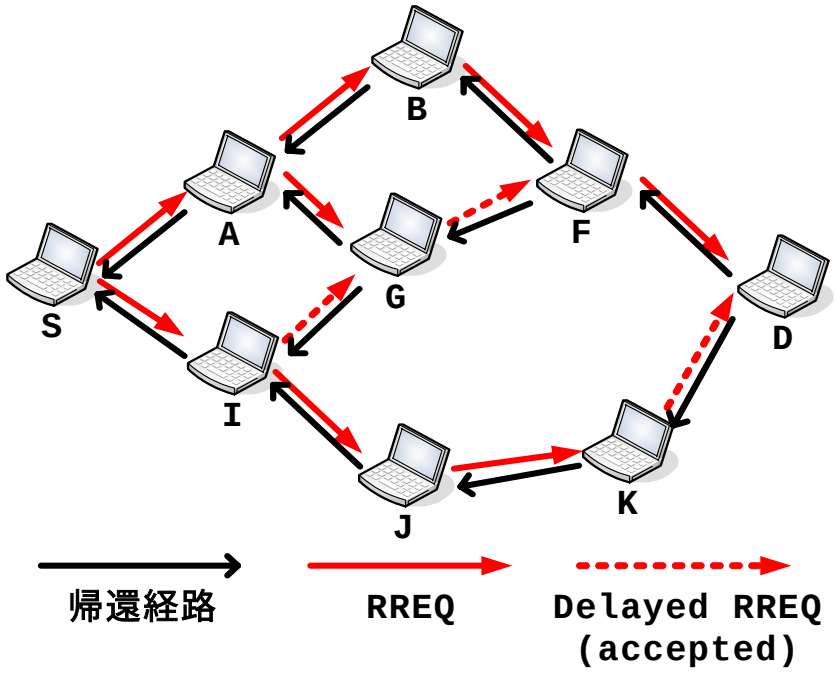


図 2.2.8 RREQ プロセス

B. Process of Route Reply

RREQ を受信した宛先 D は送信元 S へ RREP を送信する. **図 2. 2.9** のように D は複数受信した RREQ に対しては複数の RREP を返す. ノード F のように帰還経路を複数持っているノードは, バイキャストにより, 複数方向へ RREP を投げる. このとき, ノード F は RREP に追加された jointcount フィールドの値をインクリメントし, RREP を転送する. またノード A のように, ルーティングテーブルに保持されている経路(nexthop=ノード B)の jointcount より Delayed RREP の jointcount の値が大きい場合は, その RREP を転送する. ノード I のように, 先に受信した RREP(from G)の jointcount の値が, Delayed RREP のそれよりも大きい場合は最初に受信した RREP 以外は転送しない. これを繰り返して複数の転送経路を作成する.

最終的にデータを転送する経路は, 各ノードのルーティングテーブルエントリにおいて最も小さい jointcount を持つ経路から順番に選択していくことで, ディスジョイントなマルチパスを作成する. 複数のエントリが同じ jointcount である場合には, 有効期限が長い経路を選択する. この規則は, 経路作成時, ならびにリンク切断時に適用されるため, 中間ノードで経路切り替えが行われる場合もある.

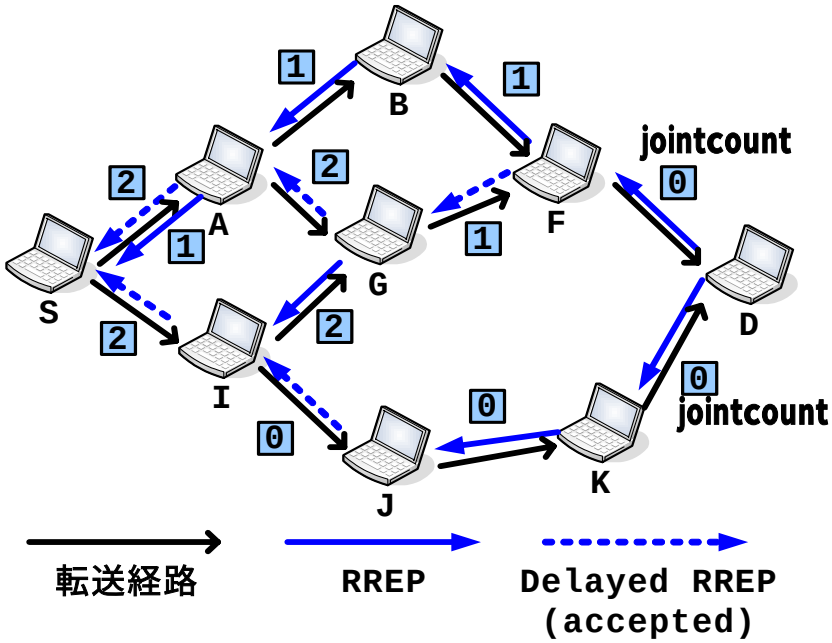


図 2.2.9 RREP プロセス

3) AODVM の問題点

図 2. 2.10 は**図 2. 2.9** によって決定した，”最も jointcount が少ない経路(ライフタイムの優先度は今は考慮外)”である．この図からもわかるように，AOMDV はﾉｰﾄﾞﾃﾞｨｽｼﾞｮｲﾝﾄパスを作る傾向があり，選択するﾙｰﾄがﾉｰﾄﾞ群の外側を通りやすい．これは，ｾｯｼｮﾝ数が増えてくると特定のﾘﾝｸに負荷がかかりやすいという特徴を持つ．また，外側を通りがちなため，例えば**図 2. 2.11** のように F-D 間でﾘﾝｸﾌﾞﾚｰｸが起きた際に，上流側(S 側)まで遡らないと，中間ﾉｰﾄﾞにおける転送経路の切り替えが出来ず，非効率である．

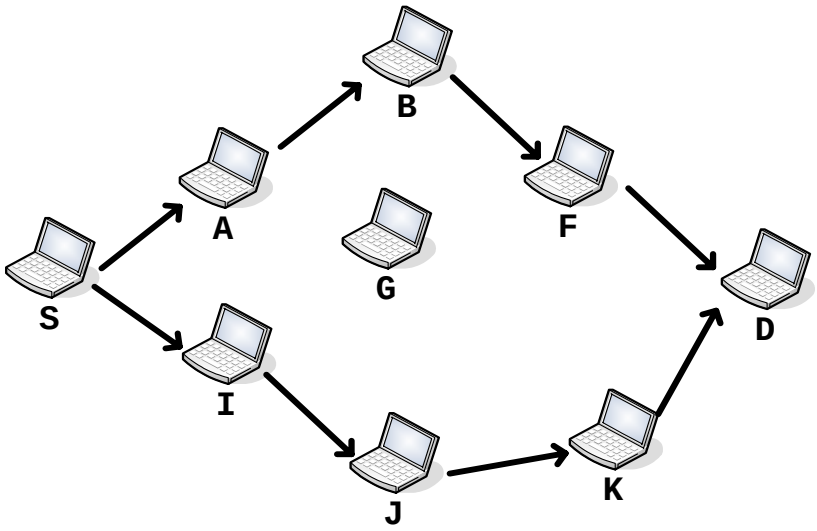


図 2. 2.10 AODVM によって決定された経路(jointocount のみ考慮)

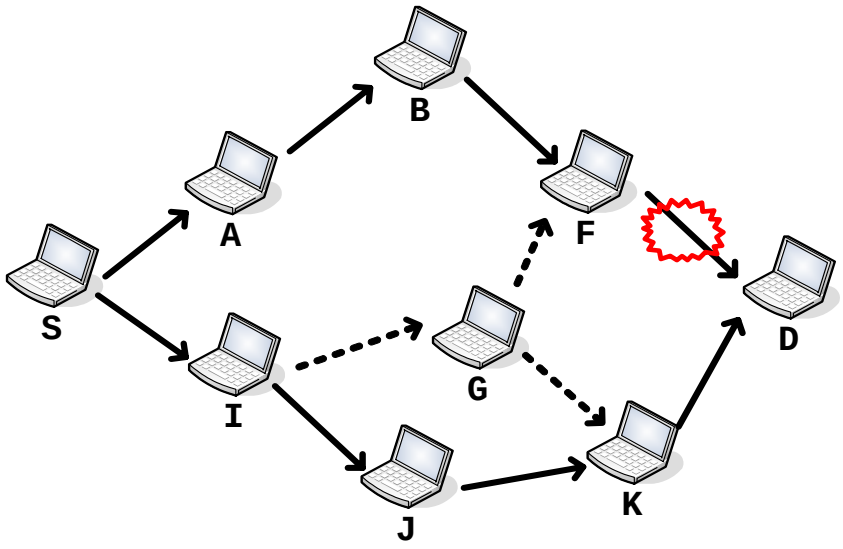


図 2.2.11 AODVM における問題点

2.2.4 Performance of Multipath Routing for On-Demand Protocols in Mobile Ad Hoc Networks[13]

DSR をマルチパス化した際にどういった経路の選び方をするとルート探索回数が減るかということについて言及している. DSR は元々多数のルートを保持し, 最初のパスが切れた際に別経路を使うということはするが, その経路を用途やライフタイムによって管理しないため, 多くのルートを保持しすぎるため, その多くの経路を活かすことができていない. このため DSR を用いてディスジョイントな経路を使った 2 つのマルチパス案を評価している.

1) DSR のマルチパス拡張

2 つのマルチパス案をプロトコル 1･2 としている.

A. プロトコル 1

宛先ノード D は最初に自身に届いた RREQ に対しては通常通りに RREP を返信する. そのルートを覚えておき, D は後に到着した RREQ のうち, ノードディスジョイントになっているものだけ選び出し, RREP を返信する. これは RREP の返信数を抑える効果も含んでいる. 送信元 S では最初のルートが切れた際には別のルートを使い, 全てのルートが切れた場合ルート探索を行なう. (**図 2.2.12(a)**)

B. プロトコル 2

プロトコル 1 では S のみが別経路を保持している. この場合, Primary ルートが中間で切れた場合に RERR パケットが送信元に返され, バックアップルートによる通信が始まる. これではすでに通信路中にあるデータパケットは廃棄されてしまう. この回避策として, 全ての中間ノード n(i) が Primary ルートに対してノードディスジョイントなバックアップルートを宛先に対して持つようにする(**図 2.2.12(b)**). D は Primary ルート上の n(i)全てにディスジョイントなルートである RREP を送信する. ただし, そのような n(i)に常にディスジョイントな経路を返すことができるとは限らないが, この研究では実現可能性を別問題として分析を行なっている.

図 2.2.12(b)において, まずリンク L(i)が切れたとすると, ノード n(i)は L(i)-L(k)間

のルートを P(i)に置き換える. P(i)が切れると n(i-1)まで RERR を返し, ルートを P(i-1)に置き換える. これを繰り返し, RERR が S に帰ると新たに経路探索がおこなわれる.

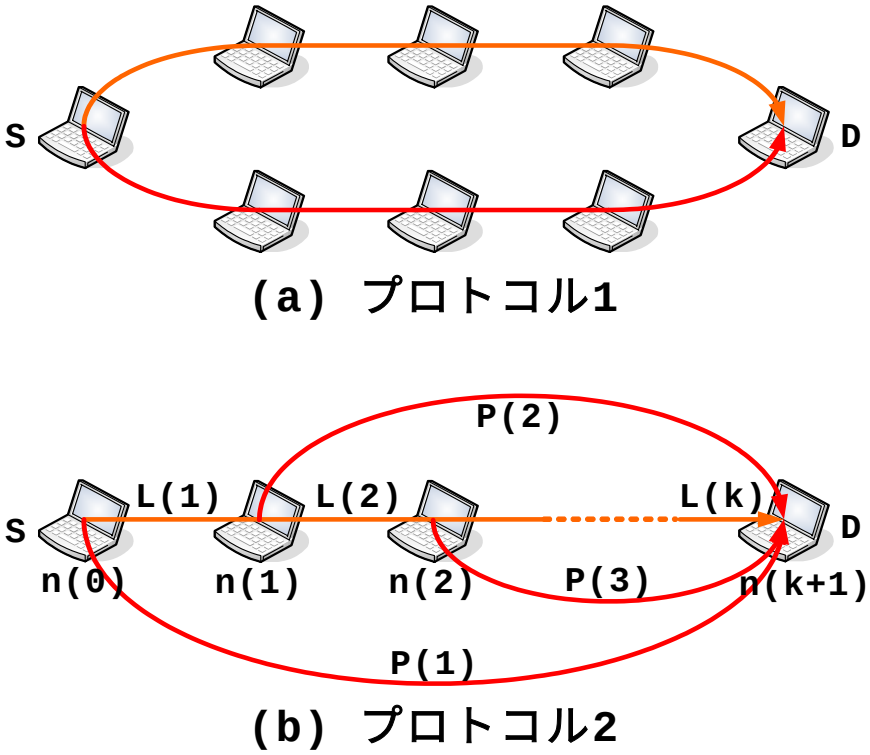


図 2.2.12 DSR のマルチパス化

2) 2 つのプロトコル性能評価

理論値評価では, 経路探索回数はプロトコル 1/2 ともにシングルパスよりも減っている. また, プロトコル 2 のシミュレーション評価では, ルーティングオーバーヘッドなどにおいてもシングルパスよりマルチパスの方が勝っているが, 遅延の面ではマルチパスが劣る. これはマルチパス化してバックアップ経路を保持した際に, その経路が長くなるとマルチパスの効果が薄れ, リンクが切れる可能性も増すからである. また, 別経路の保持数を多くしても, 経路探索回数減少の恩恵は別経路数 2~3 本程度から収束してきてしまう.

2.2.5 Fault Tolerant and Load Balancing

マルチパスを構築する上で, Fault Tolerant と Load Balancing が利点として上げられるが, その構築法によっては必ずしも効果を挙げられるわけではない.

[14]では**図 2. 2.13** のような(ノード)ディスジョイント型とメッシュ型のマルチパスルーティングプロトコルに関して,

- SPF(selective preferential forwarding):Primary/Secondary…とバックアップルートとして使う方式
- SRF(selective random forwarding):ランダムに経路を選択してデータ投げることによる負荷分散方式

を適用し, 計 4 種類のマルチパス利用法について評価している. スループット(Fault

Tolerant)に関してはﾒｯｼｭ×SPF が，負荷分散に関してはﾃｨｽｼﾞｮｲﾝﾄ×SRF が，最も効果的であると結論付けている．

また，**[15]**では従来のマルチパスにおける負荷分散の評価方法を見直し，シングルチャネル使用時に負荷分散においてはシングルパスとほとんど同じ結果を表す，と結論付けている．なぜならば，マルチパスを構築しても，その多くのﾙｰﾄが近接しているためであり，数十以上のマルチパスを作らなければ負荷分散の効果はないという．

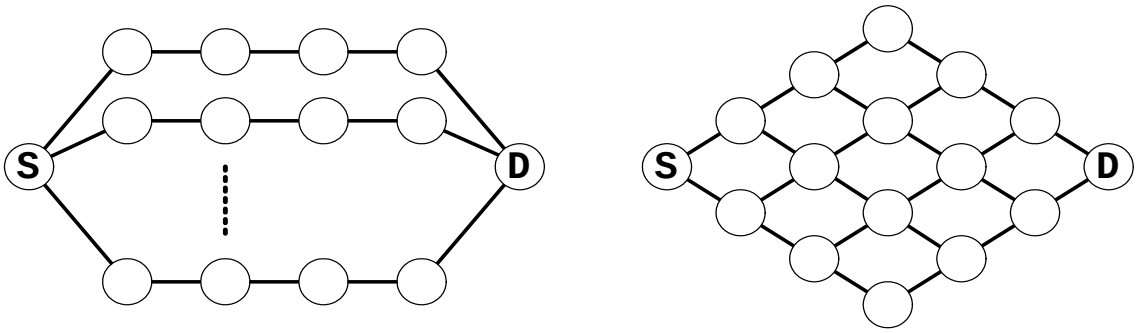


図 2.2.13 2 つのマルチパスルーティング

2.3 Implementation of MANET Routing Protocols

本節では MANET の従来の実装実験について述べる.

2.3.1 Experimental Evaluations for Wireless Multi-hop Communication

無線マルチホップ通信の評価においてはシミュレーションが主流となっているが, 実装実験による評価も近年増えてきている. その規模は室内・屋外ともに, 数ノードから数十ノードと, 徐々に大きくなってきた.

[16]では DSR をリアルタイムマルチメディアストリーミング向けに改良し, 8 ノードでの通信実験を行っている. 改良点としては

(i)Preemptive Route Maintenance: リンクの電波強度(SNR)を計測し, リンクブレークがおきそうになった際に, 代替ルートがあればそちらを使用, なければ Route Discovery を始めるという手法

(ii)Using SNR to Limit Route Discovery: RREQ はデータトランスミッションのレンジ以上に届いてしまうことがあるため, SNR のある閾値を下回ったものに関しては, RREQ の再送をしないという手法

(iii)Per-Hop Flow State Maintenance: 2.2.2 項で述べた, DSR の Flow State を使用し, パケット上にソースルートが乗ることによるオーバーヘッドを軽減する

の三点であり, 寸断のないストリーミングを実現している.

[17]では 4 つのルーティングプロトコル(AODV, APRL, STARA, ODMRP)の実装, ならびに 40 ノードでの大規模な屋外での比較実験を行っている. この実験では各ルーティングプロトコルの性能評価を実機で行っているが, これらのプロトコルに, 具体的にどのようなアプリケーションを乗せるかということについては言及していない.

[18,19]ではメッシュネットワークにより, 数十台のスタティックな無線マルチホップネットワークにおいて, スループットが最大値をとるようなメトリックの採用, ならびに評価を行っている. この実験におけるメトリックは革新的であるが, メッシュネットワークはもともと移動を想定していない無線ネットワークであるため, リンクブレークからの復帰に関しては, 重きを置いていない.

2.3.2. Kernel AODV**[20]**

我々は今回の実装実験において, 米国標準技術局(NIST)の Kernel AODV を使用した. 本節では Kernel AODV ver.2.1 についてその構成と, 動作概要を示す.

1) Kernel AODV 構成

表 2.3.1 に Kernel AODV のソースファイルの概要を示す. 表における各項は次のような意味を表している

・システムソースファイル:カーネルモジュール, netfilter の設定を行う

・AODV 処理系ソースファイル:AODV のパケット処理群が記述されている

・AODV 変数系ソースファイル:AODV の処理における参照情報の格納のための変数群が記述されている

・ユーティリティソースファイル:デバッグや上記 3 種のファイルで頻繁に使われる処理群が記述

されている

本論文末にソースコードのフローチャートを付録として添付しているので参照のこと.

表 2.3.1 ソースコード概要

ソースコード名	ソースコード概要
システムソースファイル	
module.c/h	カーネルモジュール組込に必要な関数
packet_in.c/h	netfilter における AODV パケットの受信
packet_out.c/h	netfilter における AODV パケットの送信
AODV 処理系ソースファイル	
aodv_thread.c/h	event_queue に取り込まれたパケットにより各動作に渡す
rerr.c/h	RERR 送受信・リンク削除
rrep.c/h	RREP 送受信
rrep_ack.c/h	RREP ACK 送受信
rreq.c/h	RREQ 送受信
AODV 変数系ソースファイル	
event_queue.c/h	packet_in に取り込まれたパケットを渡す queue
flood_id_queue.c/h	RREQ の ID 管理
interface_list.c/h	インターフェース情報管理関数, 取得関数
packet_queue.c/h	データパケットのキュー管理関数
rebroadcast_list.c/h	マルチキャスト用再ブロードキャスト list
rebroadcast_queue.c/h	マルチキャスト用再ブロードキャスト queue
route_table.c/h	ルーティングテーブル
neighbor_list.c/h	近隣ノードのリスト
timer_queue.c/h	タイムイベント(RREQ バックオフ, HELLO など) の queue
ユーティリティソースファイル	
aodv.h	AODV パケット定義・定数定義
utils.c/h	状態情報表示関数・型変換

2)モジュールプログラミングと netfilter

Kernel AODV では netfilter を通して実装されている. netfilter とは IP パケットの処理機構であり, Linux カーネルは受信したパケットを netfilter に渡し, netfilter はパケットの内容に基づいてその改変や消去を行う. そして netfilter による処理の後, パケットは再びカーネルに戻され, ルーティングの結果に基づき出力される.

Kernel AODV では module.c/h で init_module(),cleanup_module()を定義しており, これが通常のモジュールプログラミングにおける main(),exit()にあたる.

netfilter を通すときには netfliter フックというものを用いる. netfilter フックは netfilter における処理点である. netfilter フックの使用方法は以下の通り.

①nf_hook_ops 構造体を設定
②nf_hook_ops 構造体を引数として nf_register_hook 関数を呼び出し，netfilter にパケット処理関数を登録する
③パケットの送受信を行い netfilter によるパケット処理を実行
④ nf_hook_ops 構造体を引数として nf_unregister_hook 関数を呼び出し (cleanup_module()内)， netfilter からパケット処理関数を抹消する．

netfilter hooks を**図 2.3.1** に示す．Kernel AODV は，あるノードがパケットを受信した際に， PRE_ROUTING チェインからパケットを取得する．また，POST_ROUTING や LOCAL_OUT から送信パケットの取得，モニタリングを行っている．

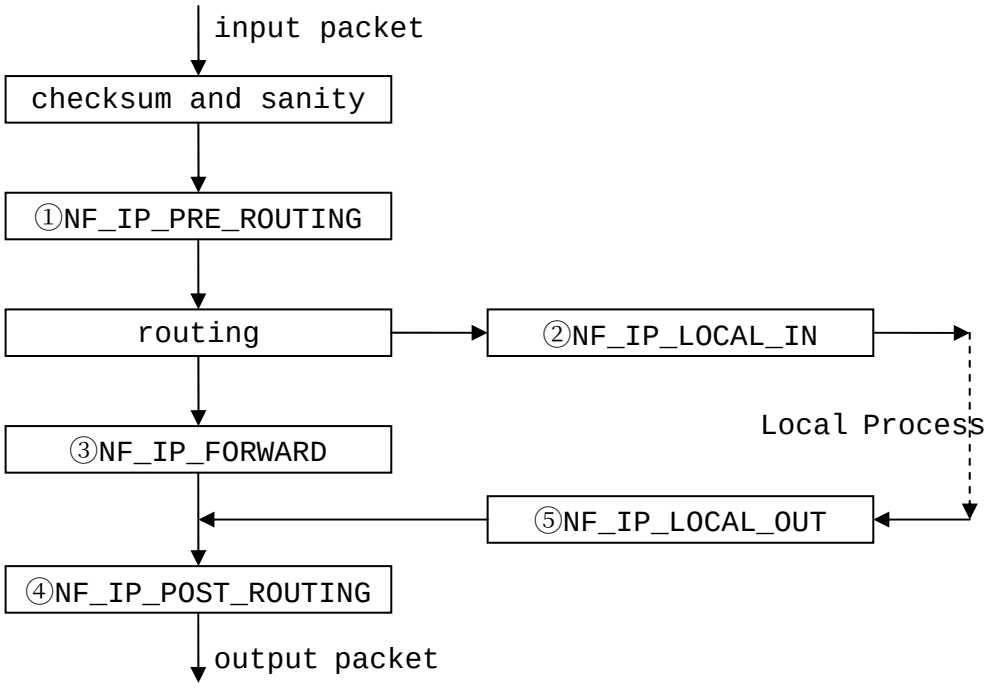


図 2. 3.1 Netfilter

3) AODV と Kernel AODV の仕様の違い
Kernel AODV は，AODV の RFC 上で規定されている Rocal Repair，Expanding Ring Search オプションを実装していない．また，アドホックネットワークにおいて，リンクブレークの検知はデータリンク層 Acknowledgement を用いる方法が推奨されているが，Linux にはデータリンク層から IP 層に情報を上げる機能はないため，Kernel AODV では，HELLO パケットによるリンクブレーク検知を行っている． HELLO_INTERVALS(1sec) * ALLOWED_HELLO_LOSSES(3 回)の間，もしくは ACTIVE_ROUTE_TIMEOUT(3sec)の間パケットの交換が行われなかった場合に，アクティブルートは無効となる．

2.4 IEEE802.11 Wireless LAN over MANET Routing Protocol

本節では IEEE802.11 無線 LAN 方式を述べると共に，MANET にて使用したときに生じうる問題点について言及する．

2.4.1 CSMA/CA

MAC 層では 2 つの異なるアクセス方法を定めている．DCF(the Distributed Coordination Function)と PCF(the Point Coordination Function)である．ｱﾄﾞﾎｯｸﾈｯﾄﾜｰｸで使われるのは DCF のほうである．

基本的なアクセス機能である DCF は CSMA/CA(Carrier Sense Multiple Access with Collision Avoidance)の機能である．CSMA は Ethernet で使われている CSMA/CD(CSMA with Collision Detection)としてよく知られている．

無線 LAN では衝突を検出できないため，CA 機能によって衝突を回避する．各端末はｷｬﾘｱｾﾝｽによって通信路の状態を確認する．ﾁｬﾈﾙがｱｲﾄﾞﾙ状態かどうかを判定するのに，IEEE802.11 では IFS(Inter Frame Space)を規定し，規定された時間以上にわたりﾁｬﾈﾙに信号が検出されない場合にｱｲﾄﾞﾙ状態であると判定する．

送信元(**図 2.4.1**，Source)では，ｷｬﾘｱをｾﾝｽした際にｱｲﾄﾞﾙ状態であれば，DIFS(DCF IFS)の間待ち，そのときｱｲﾄﾞﾙであればﾃﾞｰﾀを送信する．逆にﾋﾞｼﾞｰ状態であればﾁｬﾈﾙがｱｲﾄﾞﾙになるまで待ち，ｱｲﾄﾞﾙになってから DIFS 時間後にﾊﾞｯｸｵﾌに入る．このﾊﾞｯｸｵﾌ待ち時間はﾗﾝﾀﾞﾑな長さの待ち時間(ｾﾞﾛ～CW から決定した時間)で，直前の通信があってから一定時間後に複数の端末が一斉に送信する事態を防止している．

受信先とは違う端末(**図 2.4.1**，Others)がﾃﾞｰﾀﾊﾟｹｯﾄを受信した場合，送信されたﾊﾟｹｯﾄが自分宛でないと確認すると，NAV(Network Allocation Vector)をｾｯﾄし，ﾁｬﾈﾙがｱｲﾄﾞﾙ状態になった時点からﾊﾞｯｸｵﾌﾀｲﾑを再び減少させる．NAV は SIFS と ACK 信号送信時間を足し合わせた期間である．

ﾃﾞｰﾀの受信先(**図 2.4.1**，Destination)は受信完了後，ﾃﾞｰﾀの CRC(the Cyclic Redundancy Check)をﾁｪｯｸし，SIFS(Short Inter Frame Space)時間後、送信側に ACK を返す．ACK の受信は衝突のなかったことを示し，送信元は ACK が帰ってこない場合，ﾃﾞｰﾀを再送する．

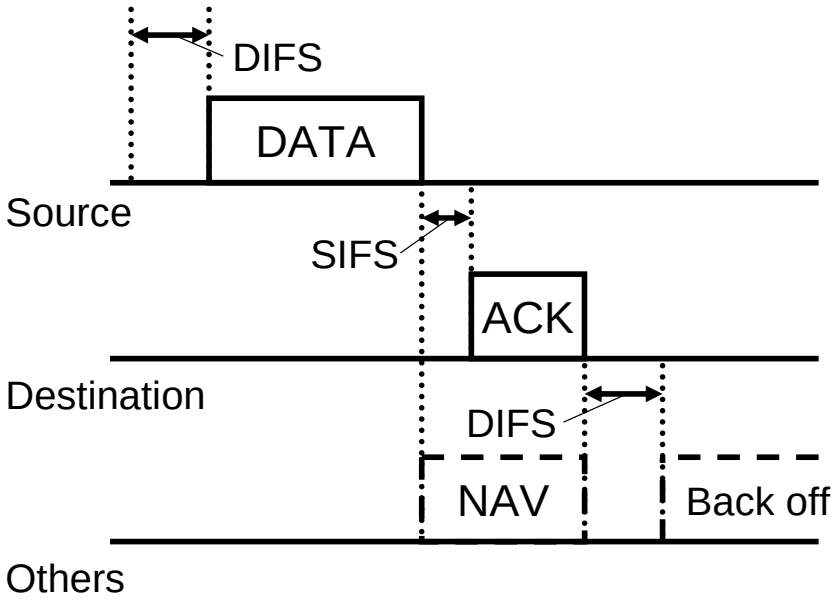


図 2.4.1 各ノードにおける送信フェーズの制御

2.4.2 Hidden Node Problem and RTS/CTS

CSMA では**図 2.4.2** で示されるような隠れ端末問題については避けることができない。これは、ノード A, B, C があった場合に, A は C と, B は C とそれぞれ通信できる範囲にあり, A と B が通信できない(キャリアセンスできない)範囲にあるとする. A から B へのデータ送信中に A を感知できない B から C へデータを送信してしまうと, パケットが衝突してしまう問題である.

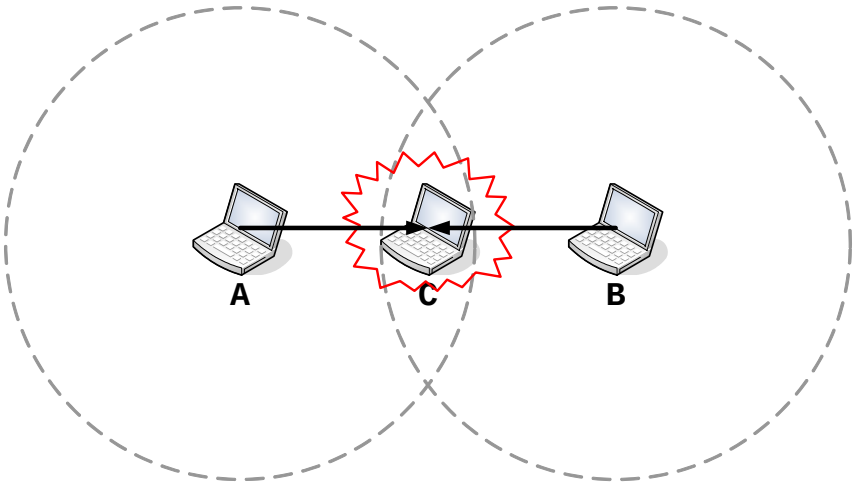


図 2.4.2 隠れ端末問題

隠れ端末問題を解決するために, IEEE802.11 には標準で仮想キャリアセンス機能である RTS(Request to Send)と CTS(Clear to Send)がある. RTS は, ACK がやりとりされるまでの時間情報(データのフレーム長情報)をもっている. RTS を受信ノード(**図 2.4.3**, Destination)が受信して, データを受信可能であれば CTS を送信する. CTS も RTS 同様の

時間情報を持っており，CTS を受信した送信ノード(**図 2.4.3**, Source)はデータを送信する．RTS and/or CTS を受信したデータの受信先でない他のノード(**図 2.4.3**, Others)は NAV をセットし，送受信が終わるまで何もせずに待つ．こうしてパケット衝突を回避する．

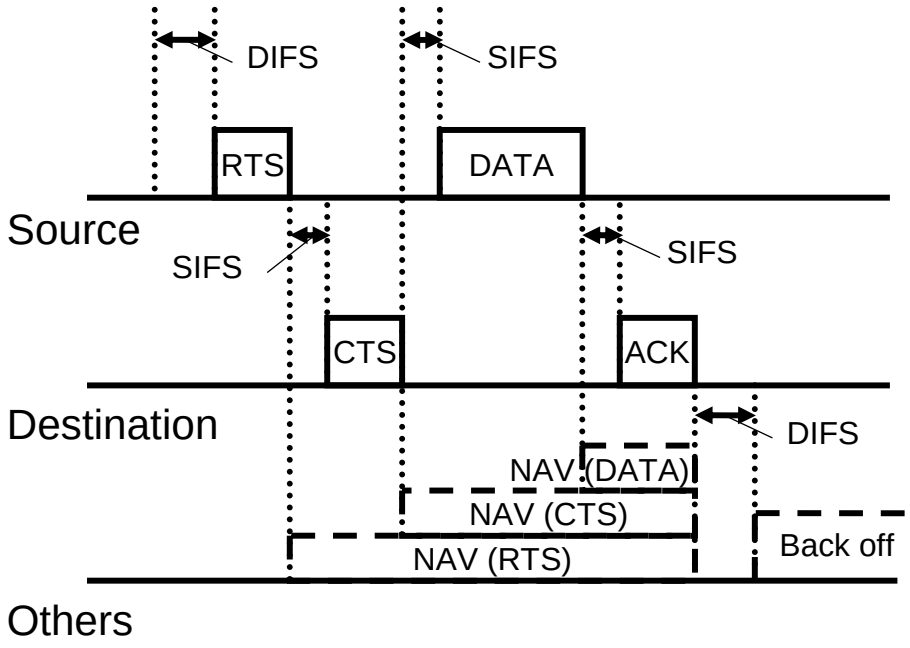


図 2.4.3 RTS/CTS 交換による衝突回避

RTS/CTS を使用しても，隠れ端末問題が解決できない場合がある．例えば，**図 2.4.4** のように，1 列にノードが並んだトポロジーがあり，ノード A がノード D へデータ送信をする場合を考える．しばらくしてノード C がノード D へデータ送信中に，ノード A がノード B へ通信しようとする状況がある可能性がある．ノード A はノード C が送信中であることを知らないので，ノード B のチャネルがアイドル状態であると認識し，ノード B へ RTS 信号を送信する．しかし，ノード B はノード C の干渉範囲にあるので，ノード B はノード A からパケットを受信することはできない．ノード C はノード A に対する隠れ端末となる．

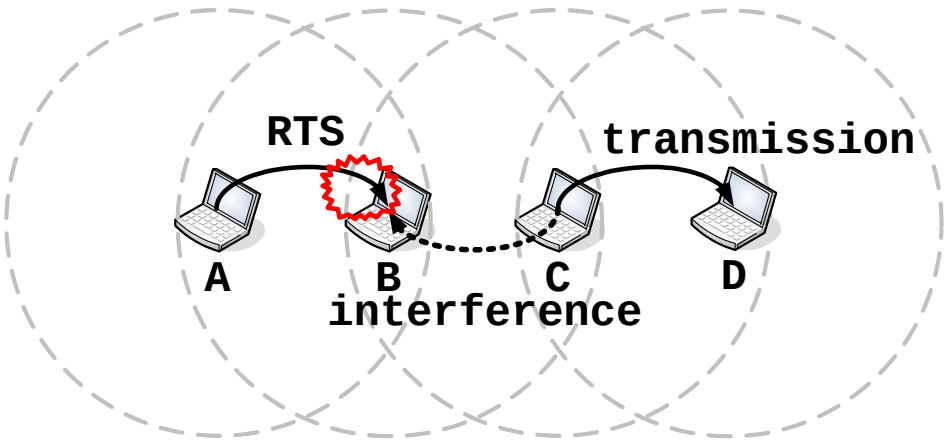


図 2.4.4 隠れ端末問題 2

2.4.3 Exposed Node Problem[21]

隠れ端末問題のほかに，さらされ端末問題(Exposed node problem)がある．さらされ端末とは，**図 2.4.5** で示すように，送信ノードの通信を感知できる範囲に入っているが，受信先ノードの範囲外(C は A の影響下だが B とは無関係)というノードである．これにより，ノード A が B に通信を始めると，C は本来ノード B での衝突の心配のない他のノード D への通信を控えてしまうこととなり，ネットワーク全体では利用可能な帯域が無駄になってしまう．この問題はアドホックネットワークにおいて深刻な問題であり，特にユニキャストパケットが増えてくると，RTS/CTS ハンドシェイクのために周りのノードが通信できない状態になってルート失効やスループットの低下の原因になる．

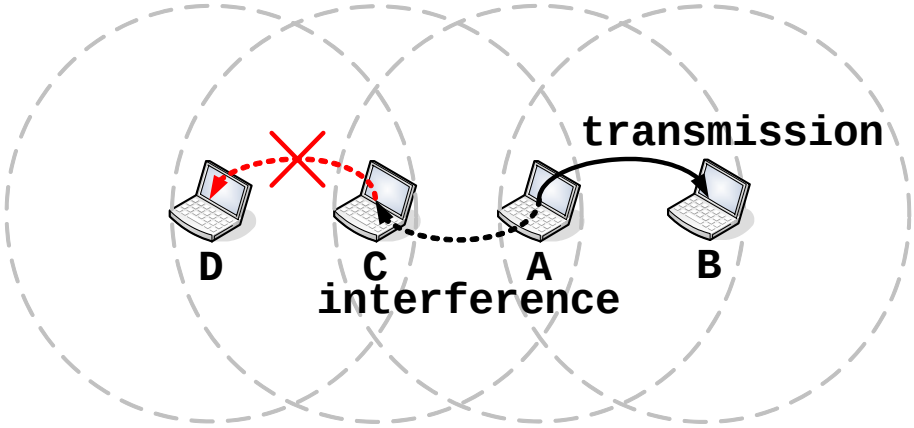


図 2.4.5 さらされ端末問題

2.4.4 IBSS Ad hoc mode vs. Ad hoc demo mode

1) IBSS Ad hoc mode[22,23]

IEEE802.11 規格では，インフラストラクチャモードで動作するときにアクセスポイント(AP)がタイミング･マスタとなり，各ステーション(ノード．以下，STA)の時刻同期を決める．このネットワークを BSS(Basic Service Set，またはセル)と呼び，時刻同期のために使われる機能を TSF(Time Synchronization Function)と呼ぶ．セル内では AP がビーコンを周期的に送信し，ビーコンを受信した STA は，AP の TSF タイマに自局のタイマを合わせる．

このマスタ-スレーブ関係がないアドホックネットワーク(以下，IBSS: Independent Basic Service Set)では，同期メカニズムが各 STA 間で働く．各 STA は制御メッセージの乗ったビーコンフレームを生成し，ネットワークの同期を行う．IBSS ではマスタ-スレーブ関係がないために，すべての STA がビーコンを生成する．各 STA がビーコン送信を予定するタイミングを TBTT(Target Beacon Transmission Time)と呼び，この情報は，最初に IBSS を構成した STA によりビーコン内に挿入されブロードキャストされる．各 STA は TSF タイマを保持し，TBTT 毎にゼロに周期をリセットしたうえで，次のような手順で動作する．

①同期のため以外のフレーム送信を中止する

②ゼロから aCWmin×aSlotTime(送受信機切り替え時間)の 2 倍までのランダム待ち時間を計算する

③ランダム待ち時間を，バックオフアルゴリズムと同様に減少

④ランダム待ち時間がゼロとなる前に他の STA からビーコンが届いた場合，その受信 STA の TSF タイマよりもビーコンのタイムスタンプの方が新しければ，受信 STA はビーコンの送信をキャンセルして TSF タイマをビーコンのタイムスタンプに同期させる
⑤ランダム待ち時間内にビーコンが届かず，期限切れとなったら，ビーコンを送る

以下，**図 2.4.6** に IBSS でのビーコンの送信を示す．この IBSS Ad hoc mode は BSS の同期が必要であるため，ネットワークへのノードの参加･離脱が激しい MANET 上での利用は適していない．

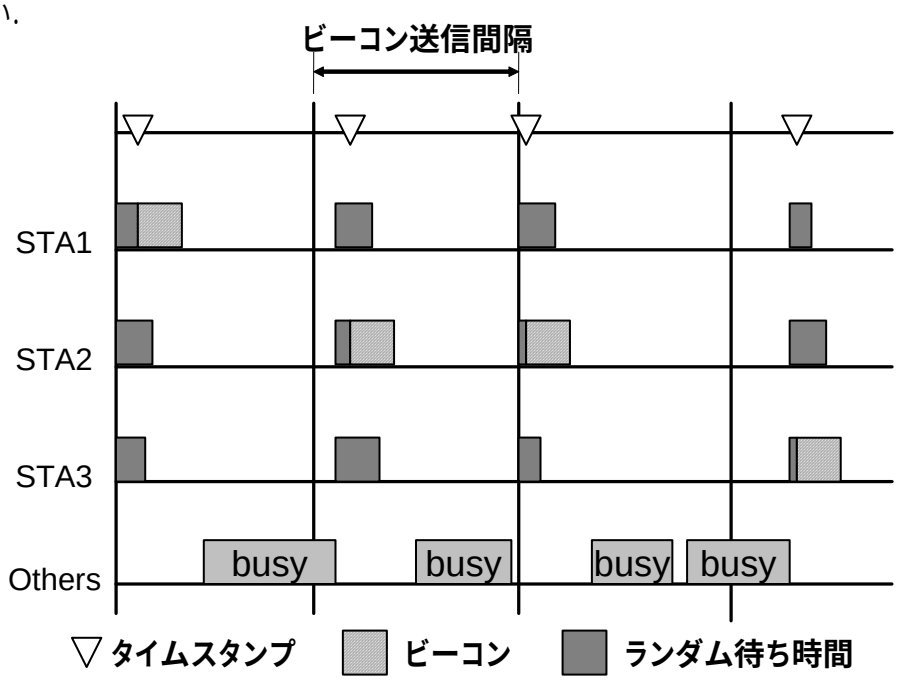


図 2.4.6 IBSS におけるビーコン送信

2) Ad hoc demo mode

Ad hoc demo mode は IEEE802.11 規格である IBSS と互換性はない，Lucent 社が独自にサポートしているアドホックモードである．Demo mode では互いにキャリアセンスはするが，ビーコンによる同期で BSS を構成することなく，同一チャネル同士であれば通信を開始できる．MANET において，このアドホックモードはうまく働くが，不特定のノードが接続できる恐れがあるため，BSS に比べセキュリティ面では劣る．

2.4.5 Gray Zone Problem[24][25]

AODV は実装上，データ送信のリンクを確保するために定期的に"HELLO"パケットを送信する．しかし，この HELLO パケットのブロードキャスト送受信により，隣接ノードを検知することが出来るが，データパケットの交換が出来ない範囲がある．この範囲を Gray Zones と呼ぶ．（**図 2.4.7**）

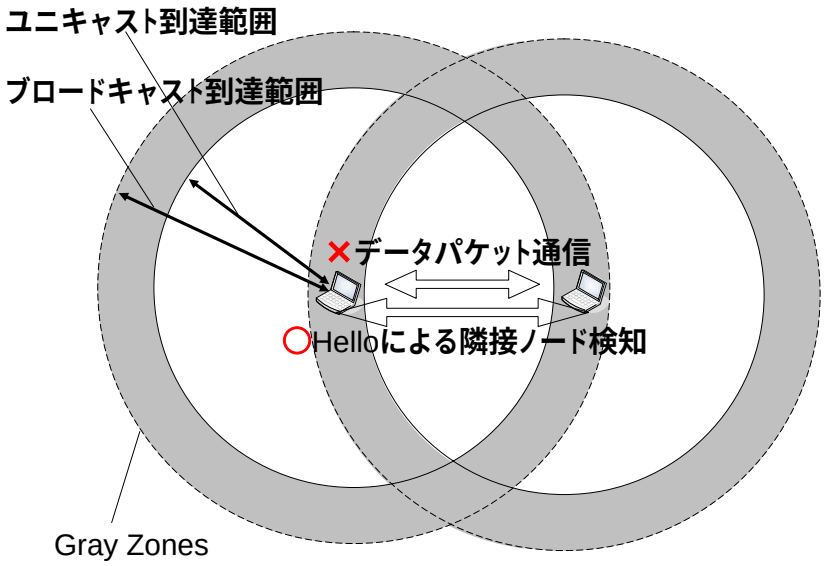


図 2.4.7 Gray Zones

この Gray Zones の原因としては以下のことが原因となっている.

(i)送信レートの違い

IEEE802.11b ではデータパケットは高い送信レート(最高 11Mbps)で送られるのに対し, ブロードキャストパケットはより低いレート(通常最高 2Mbps)で送信される. より低いレート送信の方が, より遠くへ通信が届く. これは IEEE802.11b 規格において, 偏重方式が異なることに起因する.

(ii)ACK がない

ブロードキャストには明示的な ACK がない. このため, HELLO メッセージによって維持されるリンクは双方向リンクを補償しておらず, 片方向リンクになる可能性がある.

(iii)パケットサイズの大きさ

HELLO メッセージはデータパケットに対して小さい. 小さいパケットは, 大きいパケットと比べてデータ誤りを起こす可能性は低い. また, コリジョンを起こす可能性もより少ない. より不安定なリンクでは, データパケットよりも HELLO メッセージの方が届きやすいといえる.

(iv)リンクの変動

リンクのクオリティが変動しやすい場所では, 通信が不安定になりがちである. このようなリンク上で HELLO メッセージがやり取りされると, 2 つのノード間での通信が可能かどうかを正しく反映できなくなる. これにより, 安定した長いルートがあったとしても, 不安定な最短ルートを選んでしまう可能性がある.

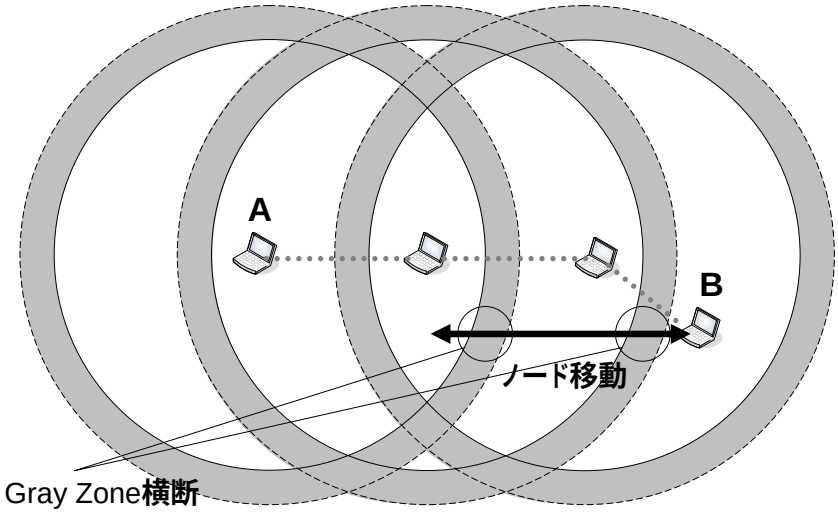


図 2.4.8 移動ノードシナリオにおける Gray Zone 問題

図 2.4.8 のようなシナリオがあった場合，A と B が通信している際に B が移動すると，Gray Zone を横断する際に A とのコネクションが切れてしまう．

このようなシナリオをもとに，Gray Zones による効果を減らす 3 つの方法を提案し，評価している．

(i)隣接ノード情報の交換

HELLO メッセージに自分の持っている隣接ノード情報を載せて送信する．HELLO を受信したノードは，送信元ノードに対し，そのリンクが双方向かどうかを知らせることができるようになる．この手法は問題(ii)を解決するが，(i)は解決しない．

(ii)N 回連続の HELLO

N 回連続での HELLO を受け取った場合，そのリンクは安定しているとする．N は2か3が望ましいが，今回の実験地としては3がとられている．この手法は問題(iv)を解決するが，(i)を解決しないうえ，レイテンシを増やしかねない．

(iii)制御パケットの SNR Threshold

IEEE802.11b のドライバから読み出したシグナルクオリティをつかって，隣接端末を決定する方法である．到着した制御パケットである閾値よりも低いものに関しては破棄するものとする．これにより不安定なリンクを排除し，ブロードキャストとユニキャストの不公平性を埋める．この実験では 8dBm を閾値として，これより下回ったものは廃棄とする．

これら三手法を AODV-UU に適用した結果，パケット到着率が，上から SNR アプローチ，Hello アプローチ，隣接ノードアプローチの順に改善されている．

3. Proposal Technique

3.1 Motivation and Protocol Design

提案手法は， 2 章の従来手法をふまえて考えると，

- 経路選択がホップカウントに制限されない
- リンク/ノードディスジョイントに制限されず，多くの経路を作成しておく
- 中間ノードでリンクブレークが起きた際に別経路に切り替えたい
- ノードディスジョイント型よりも，メッシュ型に近い経路の構築
- ただし，ノードディスジョイント経路を構築すべき場合も考慮に入れる
- アプリケーションへの影響の実装による評価

といった目標が考えられる．これらを実現するために，以下のようにルーティングプロトコルを提案する．

- 提案手法では複数経路を従来より多く作成・保持し，経路探索数を減らすことで経路寸断時の遅延を低減させる
- ソースルートリストを用いることで経路ループの回避や複数メトリックへの拡張性を与える
- これらを実装に反映させ，ストリーミングアプリケーションを実装システム上で動作させる

3.2 Proposal Technique

3.2.1 Overview

提案手法では．従来のマルチパスより多くの経路を作成し，リンク切断が頻繁に発生しても経路切り替えできる機会を増やすことによって，経路探索数を減らすことを目標としている．

RREQ/RREP にソースルートリストフィールドを追加する．ソースルートリストフィールドには RREQ/RREP を転送する際に，自身のアドレスを挿入する．これにより，ソースルートに自身のアドレスが挿入されているかを確認することでループを防いだり，ソースルートを比較することでノードディスジョイントパスのようなディスジョイントパスを作成し，経路構築ルールとしてさまざまなメトリックにより経路を作成したりすることができる．

3.2.2 Route Discovery Extension

1) Route Discovery Phase

A. Process of Route Request

図 3.2.1 において提案の動作を示す．送信元 S は通信を行う際に RREQ をブロードキャストする．ある中間ノードが初めて RREQ を受信した場合，帰還経路をルーティングテーブルに記録し，RREQ には自分のアドレスをソースルート情報として記録する．その中間ノードが，別の近隣ノードから Delayed RREQ を受信した場合，中間ノードはソースルートリストを確認する．そのノードの IP アドレスが含まれていればループが起きたと判断し，即座に Delayed RREQ を破棄する．そのノードの IP アドレスが含まれておらず，ホップ数が最初に受信された RREQ のルートより大きくなければ，到着順にバックアップの帰還経路としてルーティングテーブルの next hop として IP アドレスを保持した後，破棄する．このルーティングパケットの経路構築メトリックを"Delay Metric"[26]と呼ぶ．経路構築メトリックに関しては後述する．また，RREQ によって発見され

た経路は HELLO によって構築された１ホップのルートよりも高い優先度をもつこととしている．一つの中間ノードが保持する帰還経路数は MAX_PATH 本に限られ，本稿の実験では MAX_PATH=3 とする．

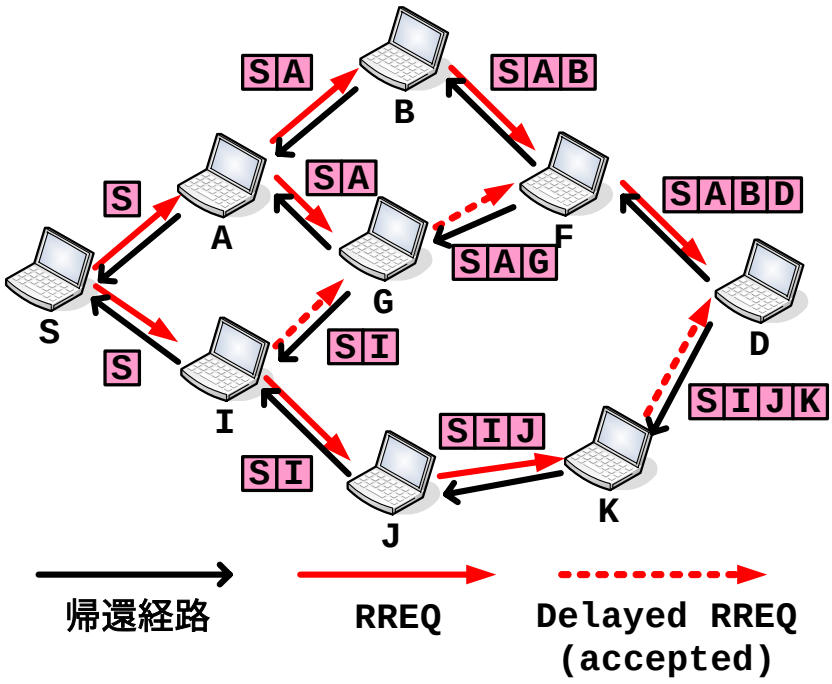


図 3.2.1 RREQ プロセス

図 3.2.2,3.2.3 にてルーティングテーブル，RREQ パケットの拡張を示す．nexthop, hopcount, route_lifetime は帰還経路毎に保持され，また，ノードディスジョイントパスの探索の際には，ルーティングテーブルにソースルートリストを書き込む．

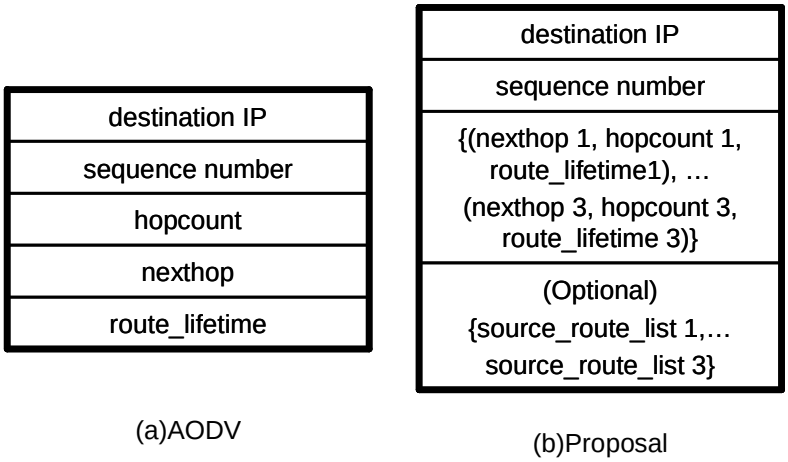


図 3.2.2　提案手法によるルーティングテーブル拡張

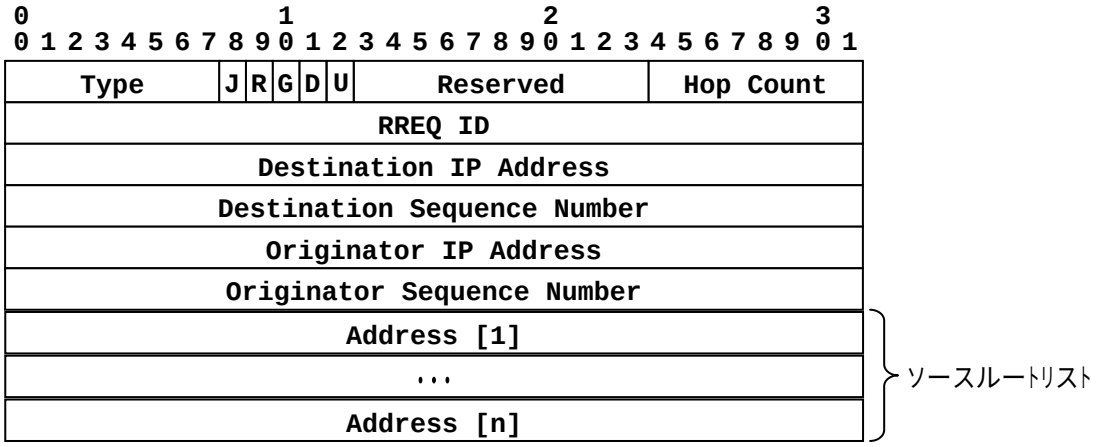


図 3.2.3　提案手法による RREQ パケット拡張

B. Process of Route Reply

図 3.2.1 にて提案の動作を示す. あて先 D が RREQ を受信すると RREP をノード S に向かってユニキャストする. 最初に受信した RREQ に対しては RREP 生成を即座に行う. ノード D で Delayed RREQ が受け入れられた場合は, バックアップルートに対して RREP を送信する. RREP パケットを受信した中間ノードは帰還経路に沿って RREP を転送するが, 帰還経路が複数ある場合にはバイキャスト送信する. 図中ノード A,I,S にて Delayed RREP は, RREQ 同様 Delay Metric によってルーティングテーブルのバックアップ転送経路を到着順に更新する. 転送経路の最大保持数 MAX_PATH は, 帰還経路と同様 3 本とする. データの転送は最初のルートが確立された時点から始まる.

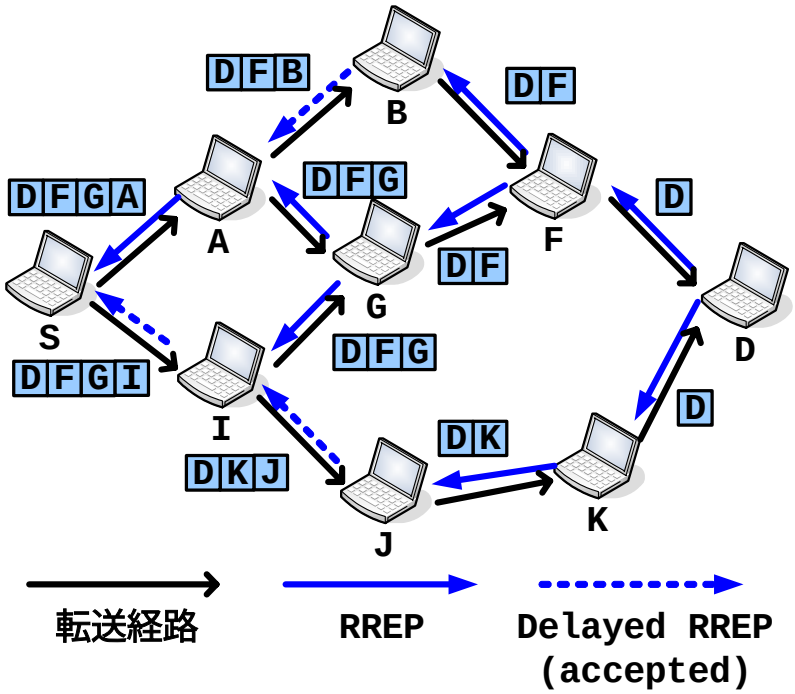


図 3.2.4　RREP プロセス

図 3.2.5 に RREP パケットの拡張を示す.

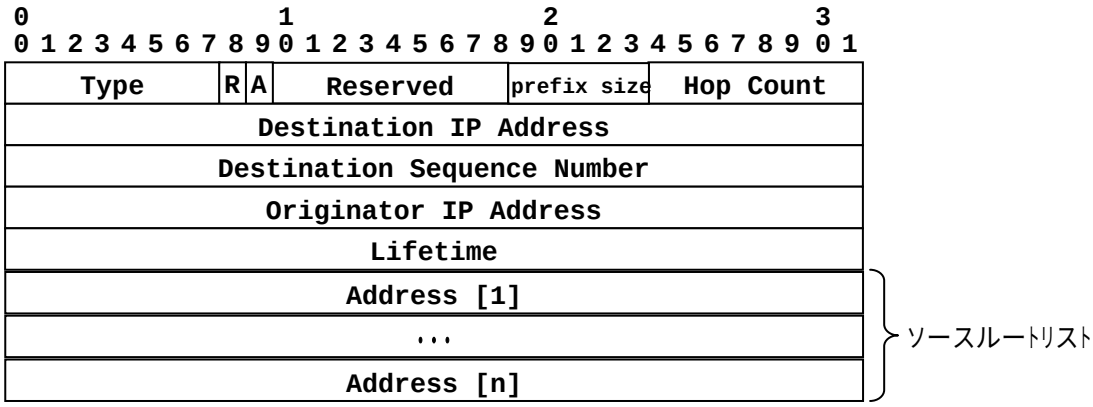


図 3.2.5　提案手法による RREQ パケット拡張

3) Metrics of Route Establishment

A. Delay Metric

それぞれのノードは RREQ/RREP を受信した場合に, RREQ/RREP が到着した順に経路を作成していく. このとき Primary 経路のホップ数より, 後から受信した RREQ/RREP に対して作成される経路のホップ数が大きい場合, 新たな経路は作成しない. 経路表に保持できる経路数 MAX_PATH を越えた場合は経路を作成しない. また, Delayed RREQ/RREP はいかなる場合も破棄される.

送信元におけるデータパケット転送の経路選択方法は, 経路が作成された順番に選択するものとし, リンク切断の際にはその順番に従って, 経路を切り替えてゆく.

B. Hop Metric

本稿では実装・評価していないが, **[26,27]**にて示されるメトリックである. それぞれのノードは RREQ を受信した場合, 以前に作成された経路の同じホップ数以下の RREQ に対する経路を作成していく. RREP を受信した場合も, 以前に作成された経路のホップ数以下の RREP に対する経路を作成していく. このとき, Delayed RREP が, 以前に作成された経路のホップ数より小さければ, RREP の再転送を行い, 上流側ノードの経路のホップ数を更新してゆく. Delay Metric 同様, MAX_PATH を越えた場合は経路を作成しない.

送信元におけるデータパケット転送の経路選択方法は, 最も少ないホップ数を持つ経路を選択するものとし, リンク切断の際にはその順番に従って, 経路を切り替えてゆく.

C. Node Disjoint Metric

Hop Metric 同様, 本稿では実装・評価していないが, **[26,27]**にて示されるメトリックである. このメトリックはルーティングテーブルにソースルートを記述するオプションを使用し, 複数経路を同時利用する際などに用いられる.

RREQ に関しては Delay Metric 同様に来た順番に更新してゆき, パケット上のソースルートをルーティングテーブルにも記載していくが, この際, ルートエントリにホップ制限は設けない. 宛先 D が RREQ を受信すると, RREP を返すが, Primary ルートに関してはユニキャストで返送,

それ以降のルートに関しては，完全に Primary に対してディスジョイントであればユニキャスト，その他はバイキャストするものとする．このユニキャスト・バイキャストに関しては Reserved フィールドから 1 ビット借りてフラグをつけるものとする．RREP がすべて送信元 S に返った時点で，S は最大限にディスジョイントなルートを使用する．

3.2.3 Route Maintenance Extension

1)RERR 処理

ノードが HELLO メッセージを近隣ノードから受信できなかった場合，ノードはリンクブレークを検知する(**2.3.3 項参照**)．もしバックアップルートがないならば，ルーティングルーティングテーブル内のルートを無効にして RERR をプリカーソルリスト上のノードに送信する．バックアップルートがあるならば中間ノードにおいて送信ルートを切り替える．経路再探索をしないことにより，パケットの遅延やルーティングパケットの送信量を減らすことが出来る．また，HELLO パケットはリンクブレーク検知だけでなく，バックアップルートのタイマを更新し，ルートの有効期限を延長させることができる．

4. Performance Evaluation

4.1 Protocol Testbed

実装環境として，我々は IBM ThinkPad 8 台，OS には Red Hat Linux 9(kernel version 2.4.20)を用いた．無線 LAN インターフェースとしては IEEE802.11b 準拠カードを oricono_cs ドライバ上で動作させた．**図 4.1.1** に今回使用したマシンを示す．

図 4.1.1 評価実験使用マシン

今回，ノード間通信を可能とする ad hoc モードには，ad hoc demo モード(**2.4.4 項参照**)を使用した．IBSS Ad hoc mode は Lucent Orinoco 無線 LAN カードでは v6.06 以降，Symbol card では v2.00 以降．PrismII cards では v0.08 以降のファームウェアでサポートされている 802.11 準拠の規格である．IBSS は新しい Ad hoc モードではあるが，default で Ad hoc モードを起動すると，Red Hat 9 上ではこちらが選択される．Linux 上に Wireless Tools**[28]**がインストールされている場合，IBSS Ad hoc mode では，iwconfig コマンドによって表示される Cell 欄に，MAC アドレスのような値を確認することが出来る．この値は，同期を始めたノードの MAC アドレスから生成される値(BSS)であり，一度リンクブレークが起きると，この値を同期させるために MANET 内のノード間通信が一切出来なくなる．実際，IBSS Ad hoc mode にて評価実験を行うと，互いに隣接するノード数が 2 台程度で，数十秒の同期遅延が発生してしまう．

```
IEEE 802.11-DS  ESSID:"kattolab_south"  Nickname:"localhost "
Mode:Ad-Hoc  Frequency:2.412GHz  Cell: 02:02:2D:2C:2B:A2
Bit Rate=2Mb/s   Tx-Power=15 dBm   Sensitivity:1/3
Retry limit:4   RTS thr:off   Fragment thr:off
Encryption key:3631-3431-33
```

図 4.1.1 iwconfig 表示結果

そこで，今回は Ad hoc demo mode を実験に使用し，純粋にルーティングプロトコルの性能を評価した．なお，これらのモードは**図 4.1.2** のようなコマンドで変更することが可能であり，Ad hoc demo mode では Cell 欄に 00:00:00:00:00:00 と表示される．

```
[foo]# iwpriv eth0 set_prot3 1 ⇒ Ad hoc demo mode
[foo]# iwpriv eth0 set_prot3 0 ⇒ IBSS Ad hoc mode
```

図 4.1.2 Ad hoc mode 切替

また，今回 IEEE802.11b を使用しているので 11Mbps まで利用可能だが，実装上の問題点としてよく知られるグレーゾーン問題(**2.4.5 項参照**)を回避するために，2Mbps 固定とした．**表 4.1.1** にテストベッド概要をまとめる．

表 4.1.1 テストベッド

マシン	IBM Thinkpad A21e×4 台，A22e×4 台
CPU	Intel Celeron 700MHz(A21e)，800MHz(A22e)
Memory	256MB(A21e)，128MB(A22e)
OS	Red Hat Linux 9 (Kernel 2.4.20)
無線 LAN カード	BUFFALO WLI-PC-11G (IEEE802.11b 準拠)
使用モード	Ad hoc “demo” mode
使用チャネル	Channel 1 (2.412GHz)
RTS/CTS	オフ
送信レート	2Mbps
WEP	なし

4.2 Performance Evaluation

本節では実装実験による Kernel AODV と提案手法との比較評価を示す. 我々は屋内において 2 つのﾄﾎﾟﾛｼﾞを使い, 3 つの実装実験を行った. 一つ目が経路回復時間, 二つ目がｽﾙｰﾌﾟｯﾄ評価, 三つ目がﾋﾞﾃﾞｵｽﾄﾘｰﾐﾝｸﾞ評価である.

4.2.1 Route Recovery latency

まず, 我々は研究室内にて**図 4.2.1** に示すように MAC ﾌｨﾙﾀﾘﾝｸﾞによって意図的にﾄﾎﾟﾛｼﾞを作り出し, Ping を用いて, 経路が切断してから通信が回復するまでの平均時間を計測した.

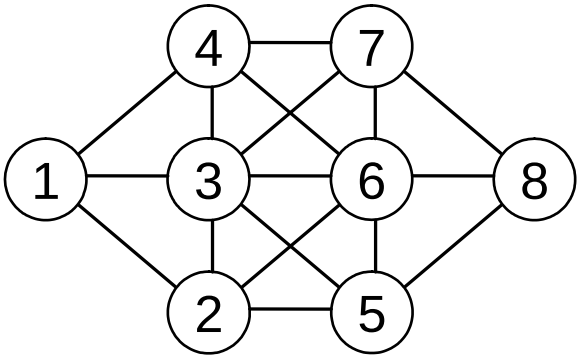


図 4.2.1 MAC ﾌｨﾙﾀﾘﾝｸﾞによるﾄﾎﾟﾛｼﾞ

図 4.2.1 において, ﾉｰﾄﾞ 1 は 512byte, 10packets/s(40kbps)という比較的穏やかなﾄﾗﾌｨｯｸ負荷でﾉｰﾄﾞ 8 に対して通信を行う. この環境下で輻輳によるﾊﾟｹｯﾄﾛｽはない. この状況下で, 通信経路中の中間ﾉｰﾄﾞをランダムにダウンさせ, 経路探索がどの程度影響を及ぼすかを観測した. **図 4.2.2** にその結果を示す. 提案手法では, ルーティングテーブルから送信ﾙｰﾄが削除された後ﾊﾞｯｸｱｯﾌﾟ経路に切り替えるため, 平均通信回復時間を短縮できていることが分かる. なお, 提案手法で経路回復に 2 秒程度かかってしまっているのは, Hello によってﾘﾝｸ切断検知を行っているためであり, 最大 HELLO_INTERVALS(1sec) * ALLOWED_HELLO_LOSSES(3 回)=3sec の間待たなくてはならないため, と考える.

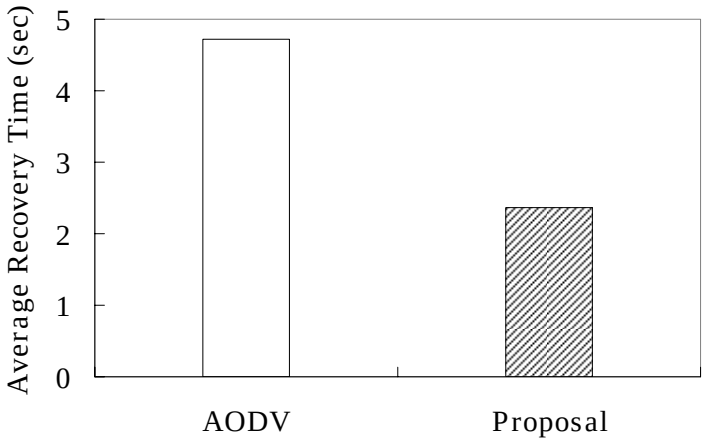


図 4.2.2 平均通信回復時間

4.2.2 Comparison of throughputs

次に，校舎内に**図 4.2.3** のようにﾉｰﾄﾞを配置し，通信状況によってﾄﾎﾟﾛｼﾞがある程度変化する環境下での場合について評価する．

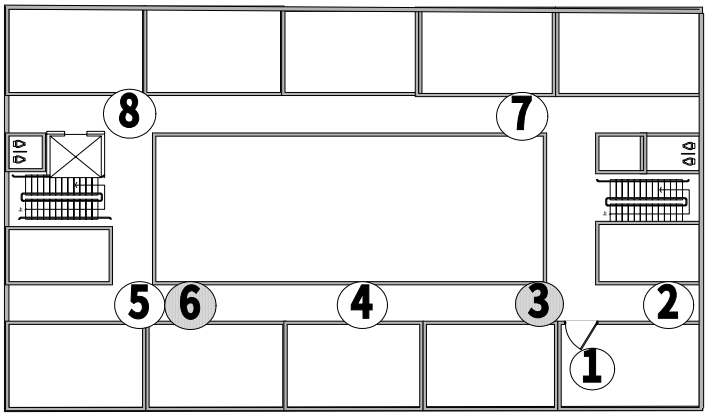


図 4.2.3 ノード配置；ノード 3 と 6 は 3 階，その他のノードは 4 階に配置

図 4.2.3 におけるﾉｰﾄﾞ 1 からﾉｰﾄﾞ 8 へ 30 秒間 TCP ｽﾄﾘｰﾑを流したときのｽﾙｰﾌﾟｯﾄをこの実験では計測した．ｽﾙｰﾌﾟｯﾄ計測には netperf**[29]**を用いた．**図 4.2.4** は平均ｽﾙｰﾌﾟｯﾄ，**図 4.2.5** はそのｽﾙｰﾌﾟｯﾄの計測値の分布である．**図 4.2.5** において 0kbps は経路探索がﾀｲﾑｱｳﾄしてしまったことを示している．平均ｽﾙｰﾌﾟｯﾄは，提案手法では平均 349kbps であり，AODV の約 1.2 倍の値を示している．

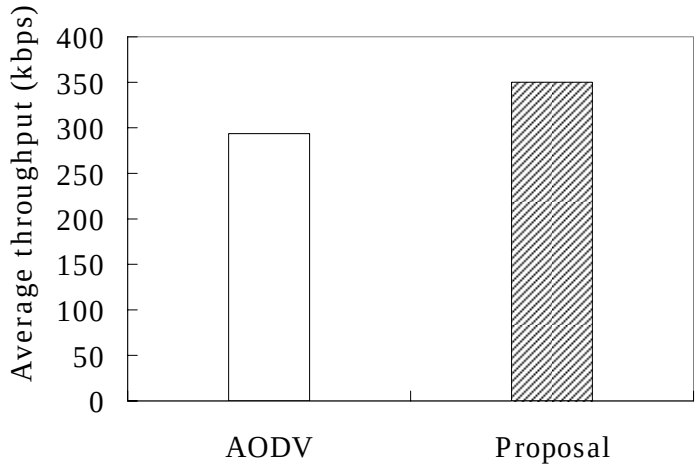


図 4.2.4 平均スループット

図 4.2.5 のスループット分布を見ると，提案手法と AODV の分布に相似性が見られる．これは，経路探索フェーズや HELLO 交換において，不安定ではあるが最小ホップであるというルートをたどりやすいことが原因であると考えられる．例えば，ノード 1 から 2,3 へのリンクは安定であるが，これらを経由するとノード 8 へは 3 ホップでたどり着き，ノード 1 から 4,5 へのリンクは不安定ではあるが，2 ホップでノード 8 へたどり着く．このためノード 1 での AODV は不安定な 4,5 を次ホップとして何度も選択し，何度も経路探索を引き起こしてしまう．我々の提案では HELLO メッセージは主にリンクのタイマ更新にしか使われないうえ，たとえ不安定リンクを選択してしまったとしても，バックアップ経路のリンクに置き換えることが出来るため，AODV ほどの影響は受けない．実際，提案手法では経路切断が起きると安定したルートに素早く収束していたが，AODV は安定ルートが構築されるまで頻繁に経路探索を繰り返していた．

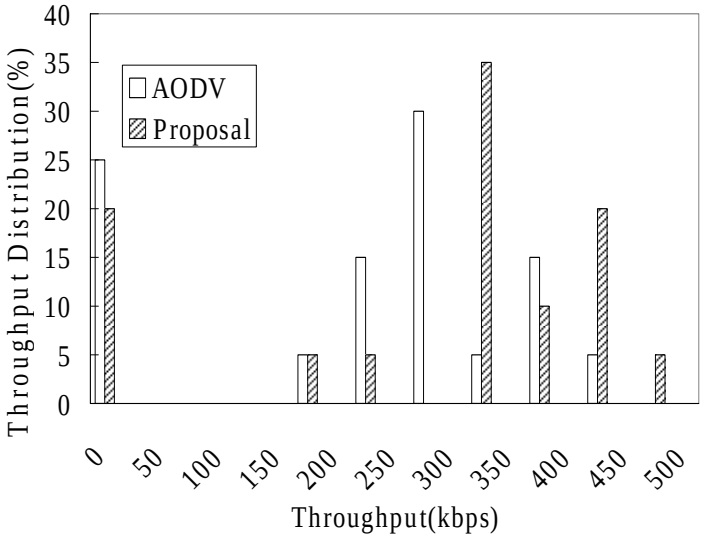


図 4.2.5 スループット分布

4.2.3 Demonstration of Live Streaming

図 4.2.3 のノード 8 に USB カメラである Logitech QuickCam Pro 4000 を取り付けて FFmpeg[30]によりストリーミングサーバとして使用，画像のコーデックには RealVideo を用いた．ビデオのビットレートは 128kbps，サイズは 320*240pixels，フレームレートは 15fps である．**図 4.2.6** にデコードされた画像を示す． RealPlayer を起動させたノード 1 からノード 8 へストリーミングを要求すると，ノード 1 は経路探索を開始し，経路確立後にストリーミングの接続要求がノード 8 へ通ることとなる．

AODV の場合，2,3 秒後に接続要求とバッファリングが終了し，ビデオは問題なく再生され，画質に関しても大きなパケットロスは見受けられなかった．しかし，少しした後に接続が切れてしまい，タイムアウトエラーが頻発した．これは先に述べたように不安定なリンクを選びリンクが切れてしまった，AODV の性質によるものであると考えられる．

一方，提案手法でも AODV と同様ビデオはストレスなく再生された．また，パケットロスもほとんどなく，タイムアウトせずに長時間ビデオを見ることが出来た．

図 4.2.6 デコードされたライブビデオ配信画像

5. Conclusion

5.1 Conclusion

第 1 章では，現在までの MANET の状況及び研究目的を述べた．

第 2 章では，今回の提案を進めるにあたり，基礎知識として必要になる研究背景及び従来手法について示した．

第 3 章では，提案手法について，その動作を述べた．

第 4 章では，提案手法の実装テストベッドを示すと共に AODV と比較しての性能評価，並びにビデオ配信デモンストレーションを行った．マルチパスルーティングプロトコルをノート PC 上に実装し，性能を評価するとともにライブストリーミング配信のデモンストレーションを行い，従来の AODV の通信性能が改善されることを示した．我々の提案では，不安定なリンクを含むルートは通信中に徐々に安定なルートへ置き換えられ，かつ，バックアップ経路に切り替わった場合にもストリーミングビデオはタイムアウトすることなくシームレスに再生された．

第 5 章は本章であり，本論文の総括について述べて締めくくる．

5.2 Future Work

我々の提案や従来のマルチパスルーティングプロトコル研究はノードの移動に対するリンク切断への耐性を高めるために考えられてきた．今回の実験では，リンク切断は不安定な電波状況によって生み出されたものであったが，今後，移動の，通信に対する影響を評価していかなければならない．

今回の研究の発展としては，パケットの受信電力(SNR)を用いての，ビデオ圧縮レイヤと協調した MAC 層における送信レート制御を考えている．この発展に関しては同研究室の野口の卒業論文にて SNR を使ったルート安定化の方策が述べられている．また，シングルチャネル環境でのマルチパスへの複数ストリーム配信は必ずしも不可分散をもたらさないため，MDC(Multiple Description Coding)[31]を用いたマルチチャネル送信による不可分散などを検討していく予定である．

References

1. Introduction

・C. E. Perkins: "Ad Hoc Networking", Addison Wesley, 2000.

[1] C. E. Perkins, P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers," SIGCOMM, 1994.
[2] IETF MANET WG
http://www.ietf.org/html.charters/manet-charter.html
[3] I.Chakeres, E.B-Royer, C.Perkins, "Dynamic MANET On-demand (DYMO) Routing", Internet Draft, October 2005.
[4] T. Clausen, P. Jacquet, "The Optimized Link-State Routing Protocol version 2," Internet Draft, August 2005.
[5] C. Perkins, E. B-Royer, S. Das, "Ad hoc On-demand Routing Vector (AODV) Routing", RFC3561, July 2003.
[6] D. Johnson, D. Maltz, Y-C. Hu, "The Dynamic Source Routing Protocol for Mobile Ad Hoc Net-works," Internet Draft, July 2004.
[7] T. Clausen, P. Jacquet, "Optimized Link State Routing Protocol (OLSR)," RFC3626, October 2003.
[8] R. Ogier, F. Templin, M. Lewis, "Topology Dissemination Based on Reverse-Path Forwarding (TBRPF)," RFC3684 , February 2004.

2. Related Works

・Samir R. Das, Charles E. Perkins, Elizabeth M. Royer and Mahesh K. Marina, "Performance Comparison of Two On-demand Routing Protocols for Ad hoc Networks," IEEE Personal Communications Magazine special issue on Ad hoc Networking, February 2001, pp. 16-28.

2.1 MANET Routing Protocols

[9] Z. J. Haas, M. R. Pearlman, P. Samar, "The Zone Routing Protocol (ZRP) for Ad Hoc Networks," Internet Draft(Expired).

2.2 Multipath Routing Protocols

[10] 西林 泰如, 甲藤 二郎, "ｱﾄﾞﾎｯｸﾈｯﾄﾜｰｸのための、マルチパスルーティング及び経路管理ﾌﾟﾛﾄｺﾙに関する研究," 平成 14 年度 修士論文, 2003.
[11] M. Marina, S. Das, "On-demand Multipath Distance Vector Routing in Ad Hoc Networks", IEEE ICNP, November 2001.
[12] S. Motegi, H. Horiuchi, and Members, "AODV-Based Multipath Routing Protocol for Mobile Ad Hoc Networks", IEICE Trans. Commun., Vol.E87-B, No.9, September 2004.

[13] A. Nasipuri, R. Castaneda, S. Das, "Performance of Multipath Routing for On-Demand Protocols in Mobile Ad Hoc Networks," Mobile Networks and Applications, vol6, August 2001.
[14] S. De, C. Qiao, "On Throughput and Load Balancing of Multipath Routing in Wireless Networks," IEEE WCNC2004, March 2004.
[15] Y. Ganjali, A. Keshavarzian, "Load Balancing in Ad Hoc Networks: Single-path Routing vs. Multi-path Routing," IEEE INFOCOM 2004, March 2004.

2.3 Implementation of MANET Routing Protocols

[16] Y-C. Hu, D. Johnson, "Design and demonstration of live audio and video over multihop wireless ad hoc networks," Proceedings of Military Communication Conference (MILCOM), October 2002.
[17] R. Gray, D. Kotz, C. Newport, N. Dubrovsky, A. Fiske, J. Liu, C. Masone, S. McGrath, and Y. Yuan. "Outdoor experimental comparison of four ad hoc routing algorithms," In ACM/IEEE International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM), October 2004.
[18] D. S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris. "A high throughput path metric for multi-hop wireless routing," In Proceedings of the ACM International Conference on Mobile Computing and Networking (MobiCom), September 2003.
[19] J. Bicket, D. Aguayo, S. Biswas, and R. Morris, "Architecture and Evaluation of an Unplanned 802.11b Mesh Network," In Proceedings of the ACM International Conference on Mobile Computing and Networking (MobiCom), September 2005.
[20] Kernel AODV 'http://w3.antd.nist.gov/wctg/aodv_kernel/'

2.4 IEEE802.11 Wireless LAN over MANET Routing Protocol

・ 松江英明, 守倉正博 監修 "802.11 高速無線 LAN 教科書,"IDG ジャパン, 2000 年
・ C.K-Toh 著, 構造計画研究所 訳, "ｱﾄﾞﾎｯｸﾓﾊﾞｲﾙﾜｲﾔﾚｽﾈｯﾄﾜｰｸ," 共立出版, 2003.

[21] S. Xu and T. Saadawi, "Does the IEEE 802.11 MAC protocol work well in multihop wireless ad hoc networks?," IEEE Communications Magazine, Vol.39, No. 6, pages 130-137, June 2001.
[22] L. Huang and T-H. Lai, "On the Scalability of IEEE 802.11 Ad Hoc Networks," ACM Mobihoc, June 2002
[23] P. Rauschert, A. Honarbacht and A. Kummert, "On the IEEE 802.11 IBSS and its timer synchronization function in multi-hop ad hoc

networks," 1st International Symposium on Wireless Communication Systems (ISWCS), Sept. 2004, pp.304 - 308
[24] H. Lundgren, E. Nordstro"m, C. Tschudin, "The Gray Zone Problem in IEEE 802.11b based Ad hoc Networks," ACM SIGMOBILE Mobile Computing and Communications Review, 2002
[25] H. Lundgren, E. Nordstrom, and C. Tschudin, "Coping with communication gray zones in IEEE 802.11b based ad hoc networks," In ACM inter-national workshop on Wireless mobile multimedia (WoWMoM), September 2002.

3. Proposal Technique

[26] Y. Sakurai, J. Katto, "AODV Multipath Extension using Source Route Lists with Optimized Route Establishment," International Workshop on Wireless Ad-hoc Networks (IWWAN), May 2004.
[27] 櫻井 祐介, 甲藤 二郎, "ソースルートリストを用いた AODV マルチパス拡張," 平成 16 年度 修士論文, 2005

4. Performance Evauation

[28] Wireless Tools for Linux
http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Tools.html
[29] The Netperf Homepage
http://www.netperf.org/netperf/NetperfPage.html
[30] FFmpeg
http://ffmpeg.sourceforge.net/index.php

5. Conclusion

[31] V. K. Goyal. "Multiple Description Coding: Compression Meets the Network," IEEE Signal Processing Magazine, September 2001.

Acknowledgements

本論文を作成するにあたり， 3 年以上に渡りご指導，ご助言を戴きました甲藤二郎教授に深く御礼申し上げます．学士４年から３年もの間，ラムダックスビルという恵まれた研究環境を与えてくださり，感謝致しております．また，修士になってからは研究だけではなく，マシン/ネットワーク管理に関して，生活に関してなど，これから先においても役に立つであろう事をたくさん学ばせていただきました．どうもありがとうございました．

また卒業されてからも，さまざまなアドバイスを頂いた高宗俊輔氏，津田健吾氏，國近洋平氏，そしてラムダックスビルでは 2 年間を共に過ごした先輩の櫻井祐介氏，後輩の小泉信也氏には，特に深く感謝致します．

さらに，3 年間同じネットワーク班員としてお互いに切磋琢磨してきた岡田陽平氏，研究班は違えど，同期としていつも励ましあってきた石先広海氏，加藤喬氏，山崎篤史氏，マシン/ネットワーク管理では頼りない私についてきてくれた楠本哲也氏，近藤裕介氏，斎藤匡志氏，また，短い間でしたが共同で研究を頑張った森井健之氏，小谷幸宏氏，野口和浩氏に心よりお礼申し上げます．

素晴らしい環境を築き上げてくれた甲藤研究室全ての皆さまにも感謝するとともに，これからますます研究室が発展するとともに，アドホックネットワークが早稲田大学の中でもっと盛り上がりを見せることを心より願っております．

最後になりましたが，研究を進めるにあたって心身ともにバックアップしてくれた家族，愛すべき友人達に感謝致します．

ありがとうございました．

2006 年 2 月 7 日

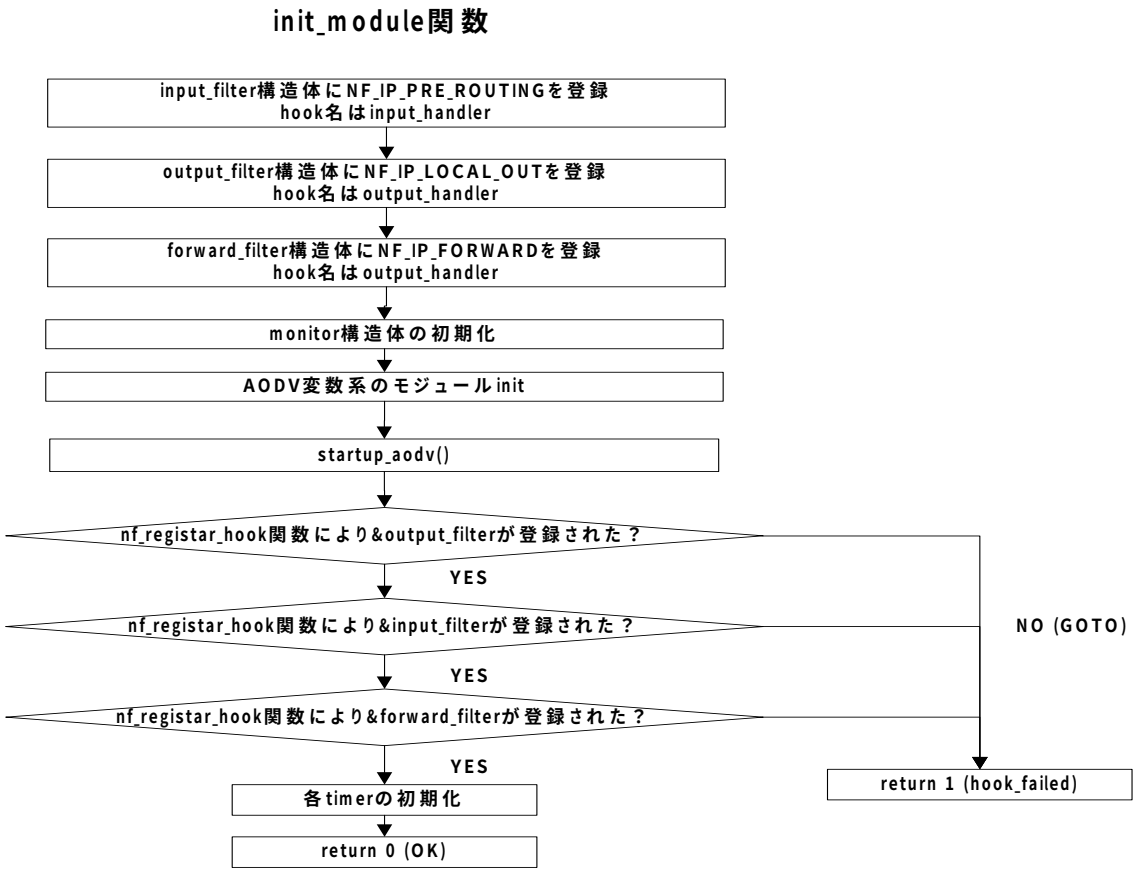
谷山 健太

A. Kernel AODV Flowchart

本付録では **2.3** 節で紹介した Kernel AODV をフローチャート化したものを掲載する.

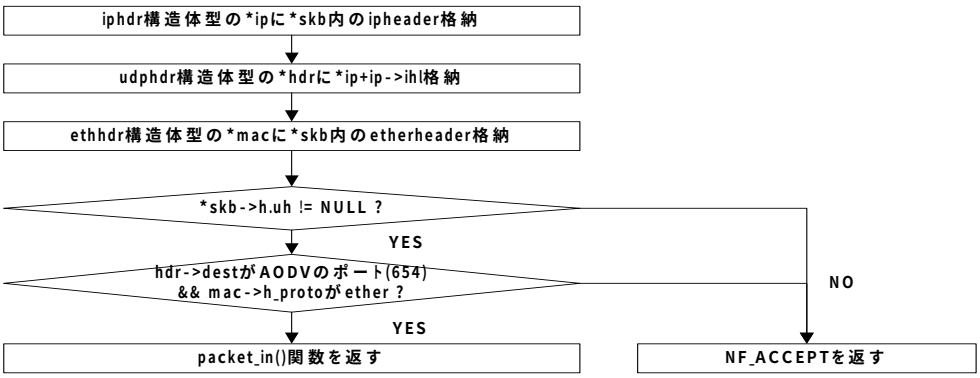
A.1 Kernel Module Functions

@module.c

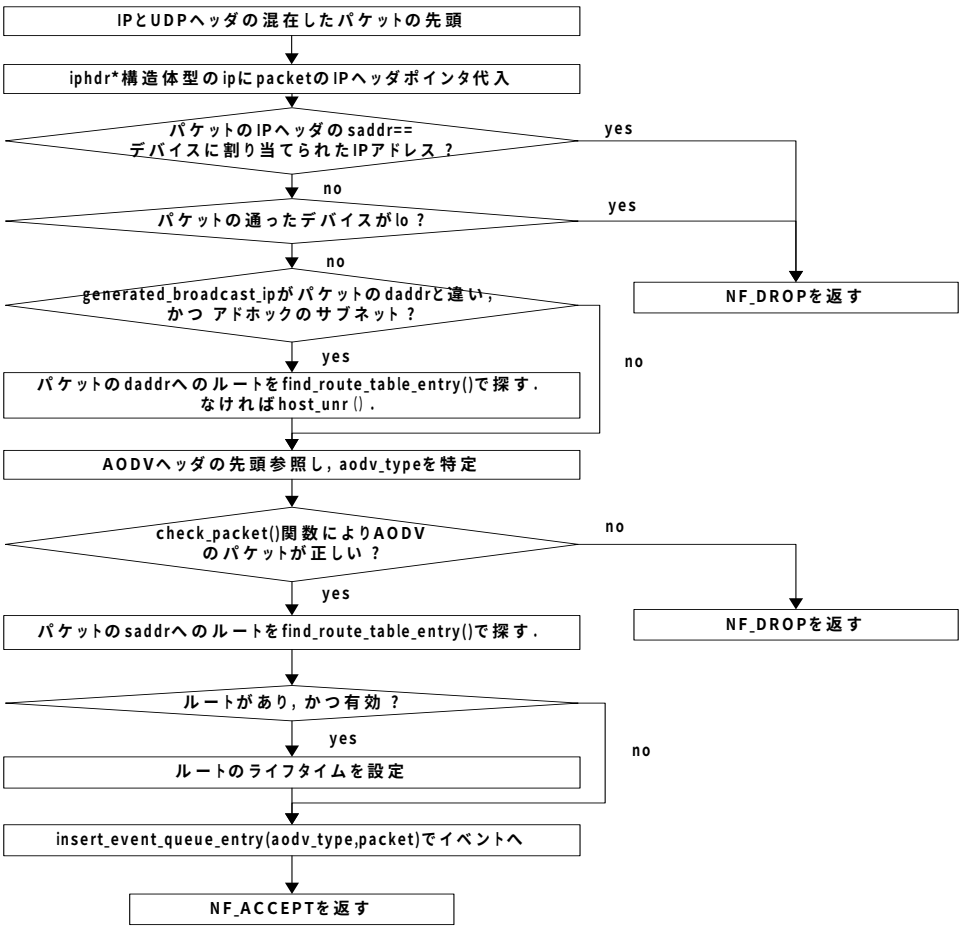


@packet_in.c

input_handler関数: hooknum, sk_buff, netを引数．netfilterがパケット受信の際に呼ばれる．



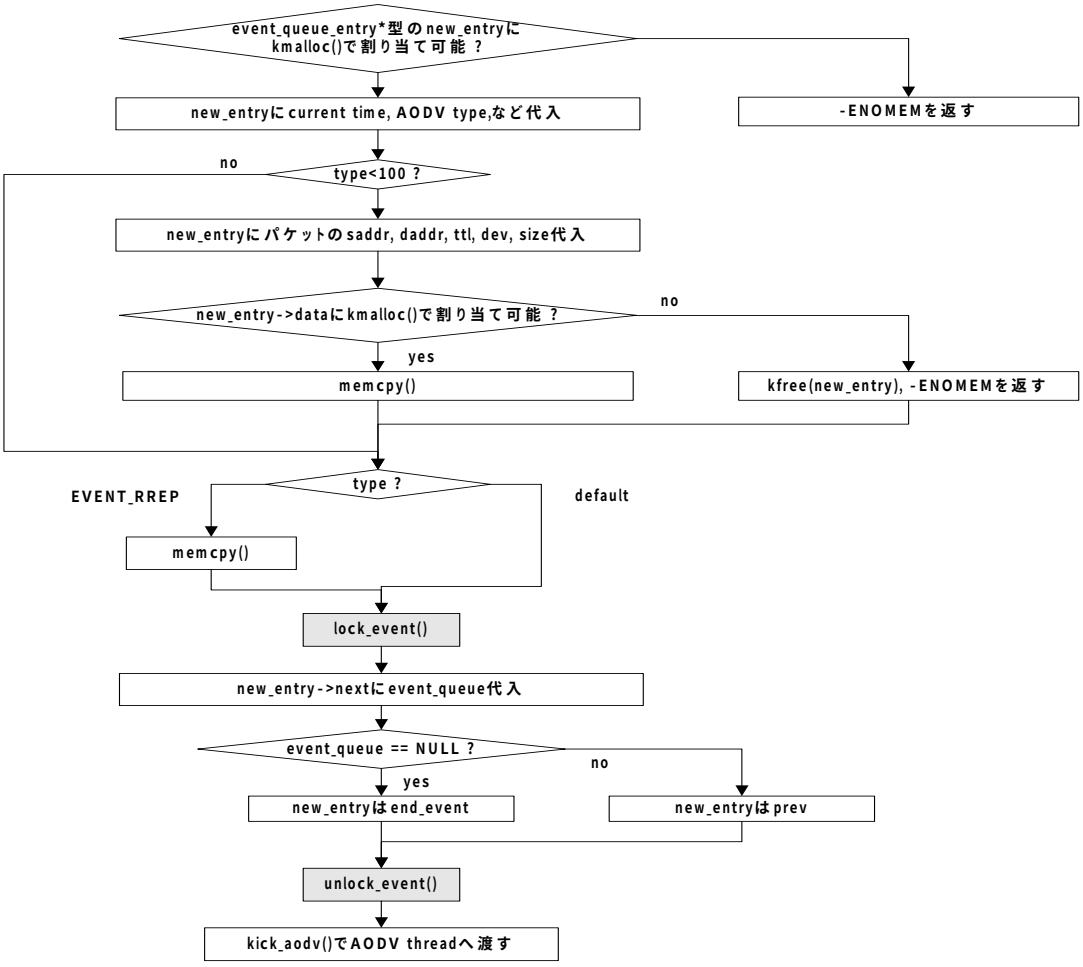
packet_in関数: input_handlerにおいてAODVパケット受信の際に呼ばれる．



A.2 AODV Event Functions

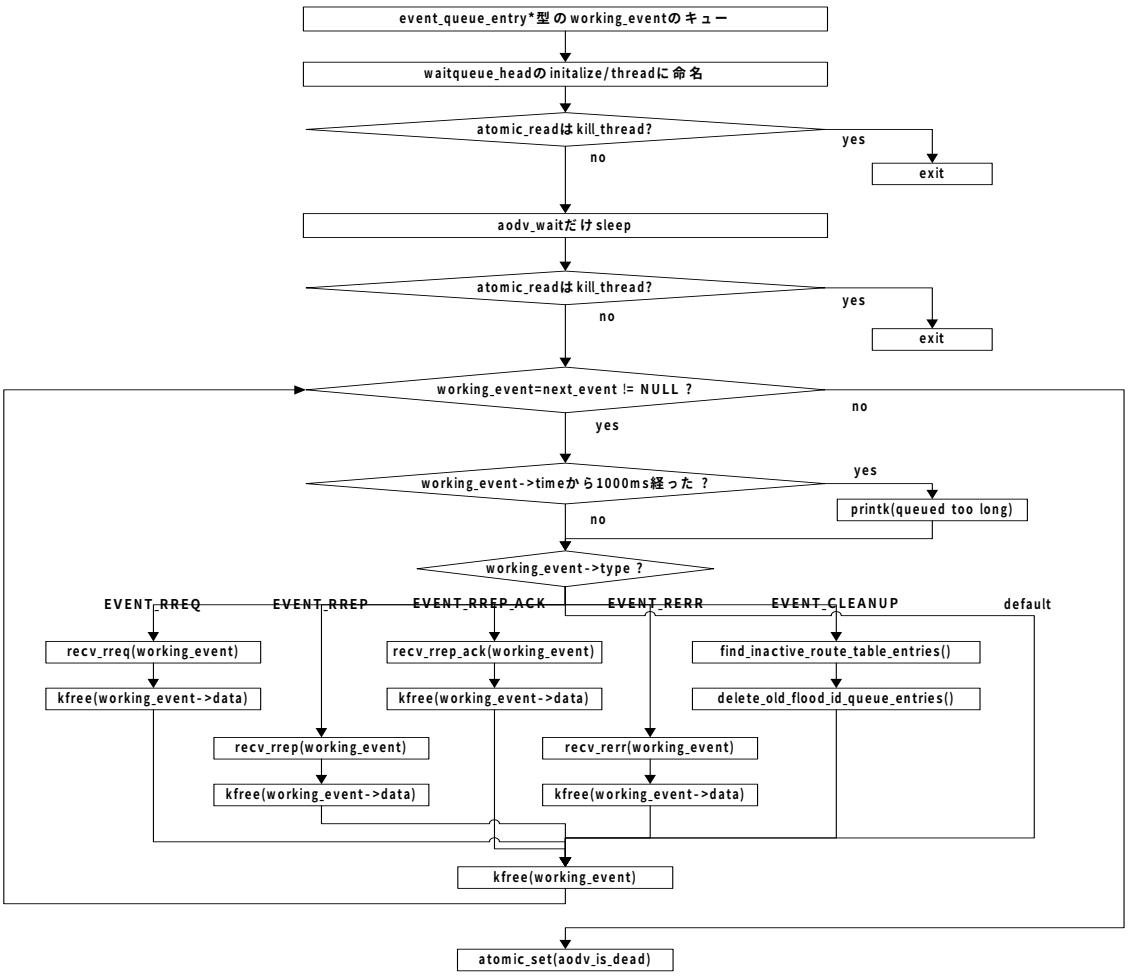
@event_queue.c

insert_event_queue_entry関数:
event queueへのキューイング. devはイベントが受信されたデバイス. dataはイベント自身.



@aodv_therad.c

aodv()関数: loopでイベントキュー内の全てデキューし, sleepに入る



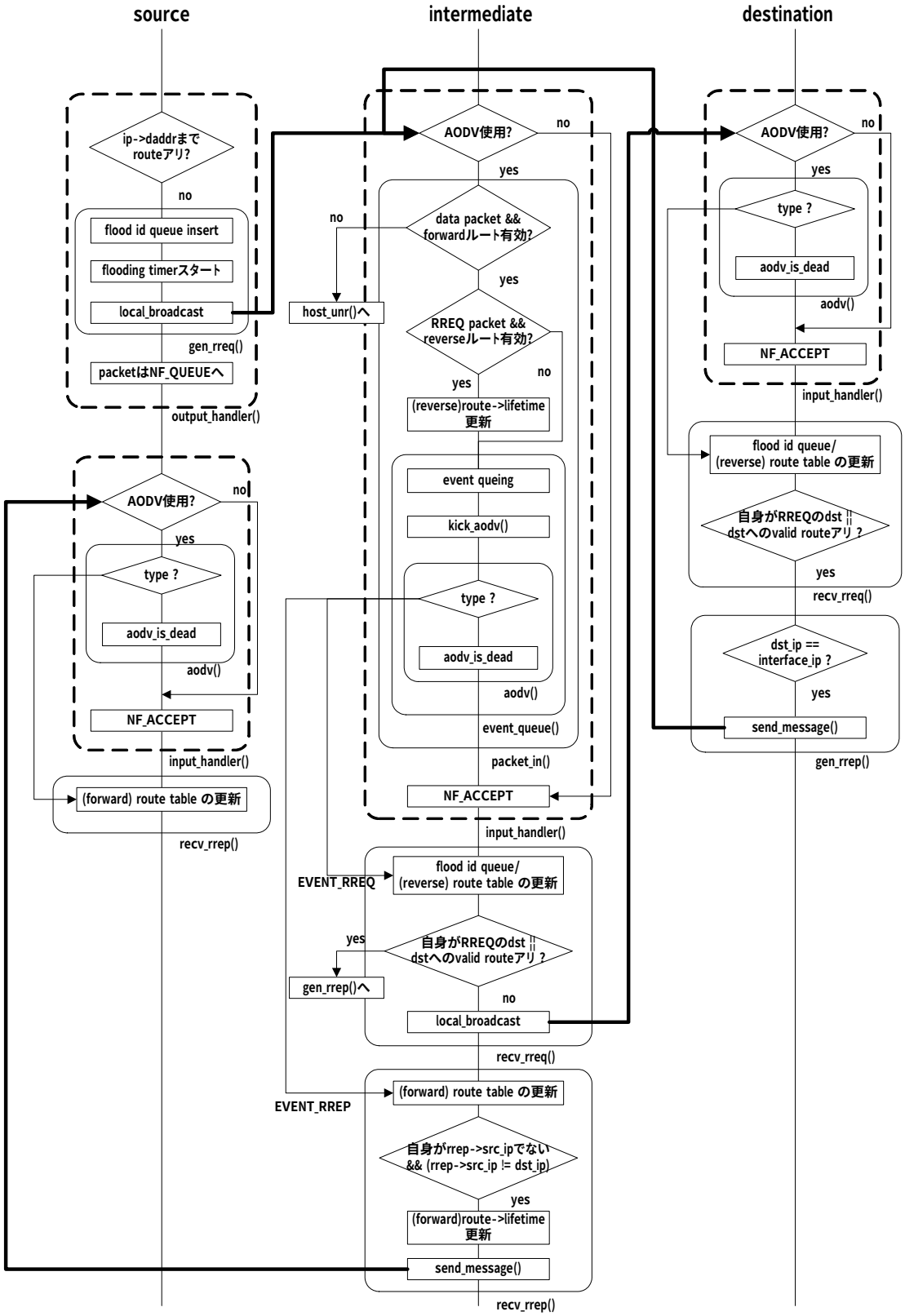
startup_aodv()関数: aodv()を始める

kill_aodv()関数: aodv()をwaitへ

kick_aodv()関数: aodv()をwakeupへ

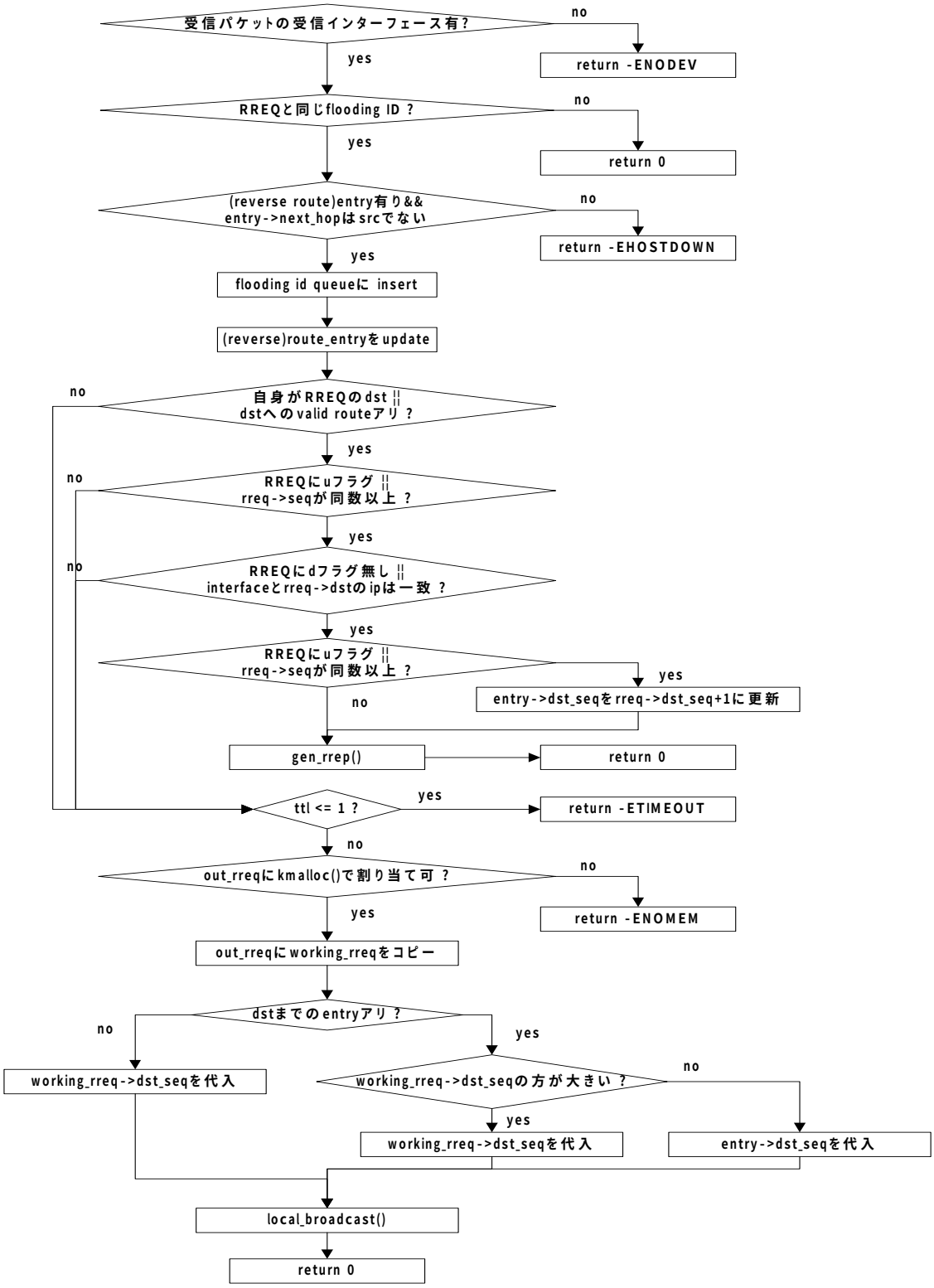
A.3 AODV Route Discovery Functions

-Process of Route Discovery



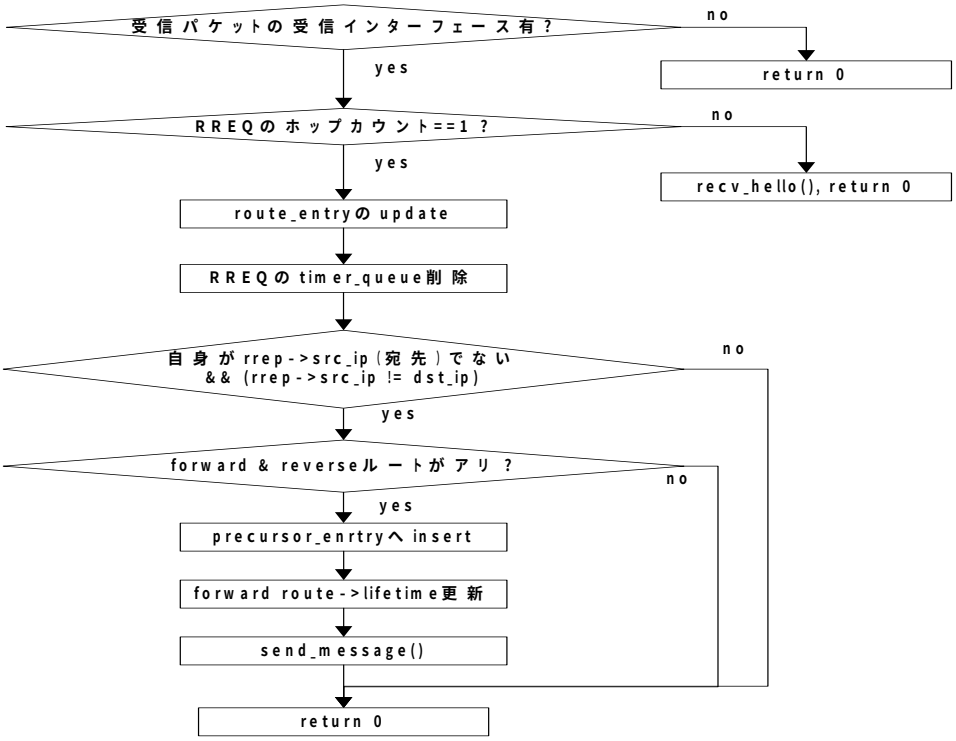
@rreq.c

recv_rreq()関数: RREQ受信時

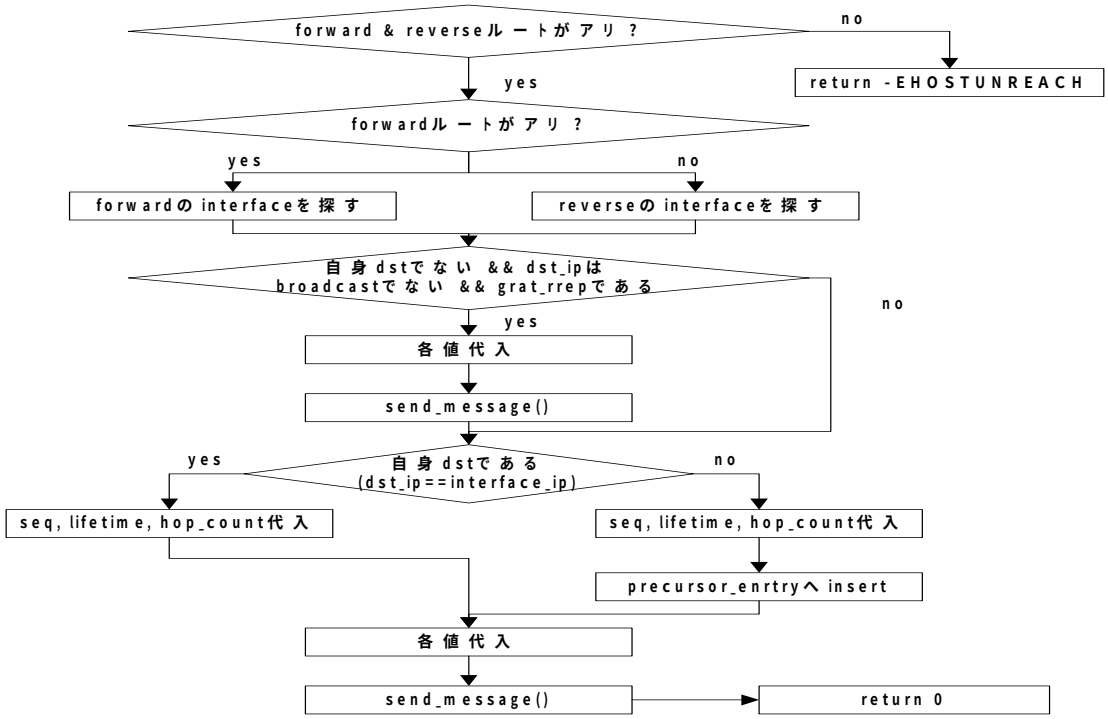


@rrep.c

recv_rrep()関数：RREP受信時

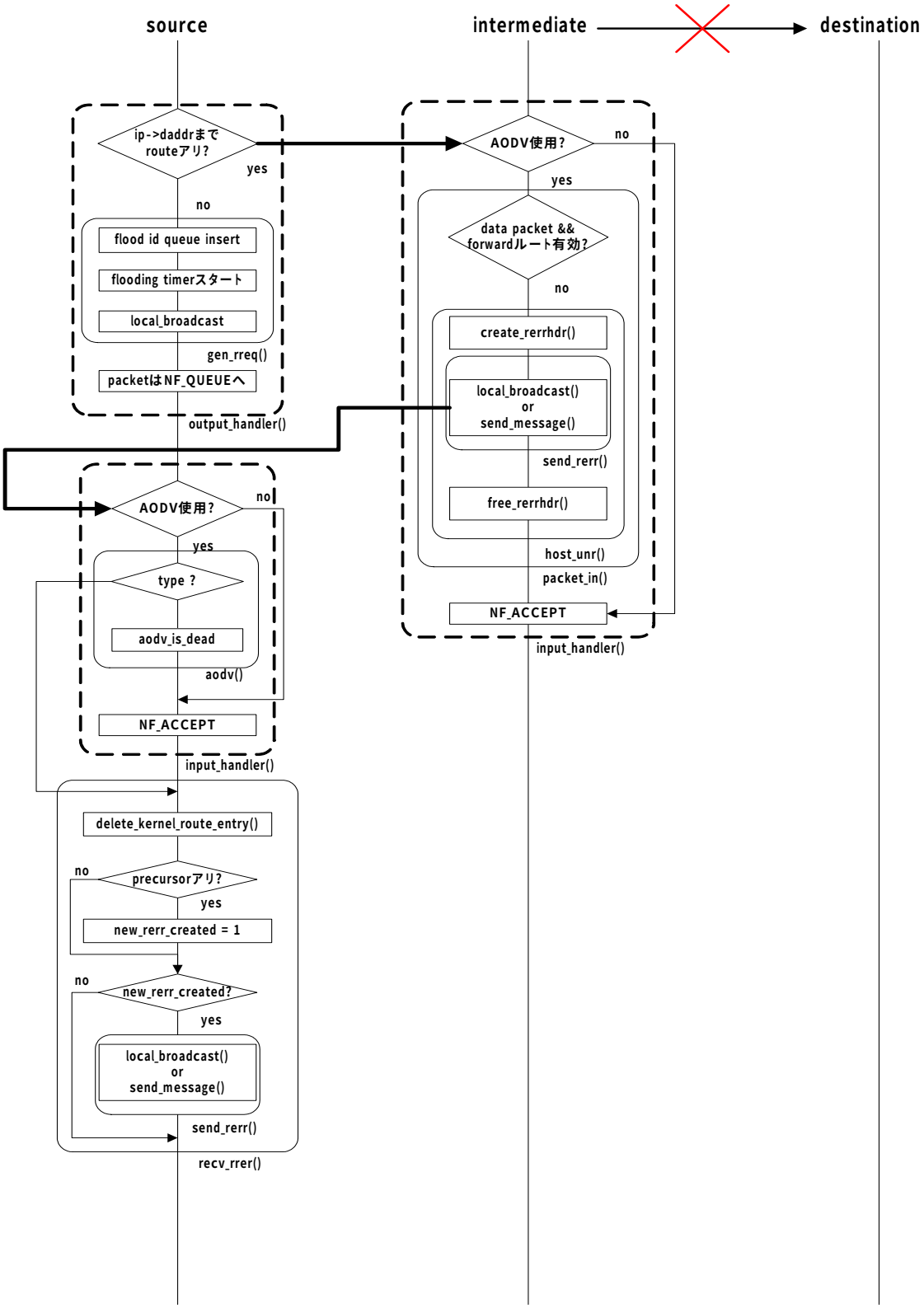


gen_rrep()関数：RREP送信時



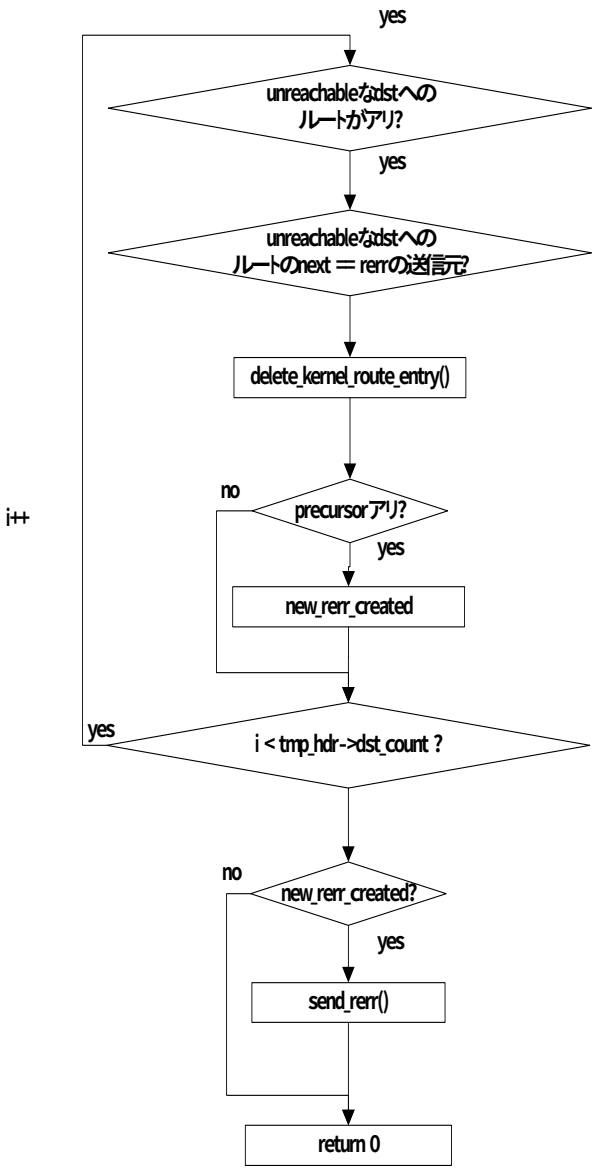
A.3 AODV Route Maintenance Functions

-Process of Route Maintenance



@rerr.c

recv_rrer()関数 RERR受信時



B. Papers

本付録では筆者が携わってきた研究について記す.

- “アドホックネットワークにおけるディスジョイント・マルチパスルーティング”
  電子情報通信学会 秋季全国大会, B-7-43, Sep. 2004

- “ソースルートリストを用いた AODV マルチパス拡張”
  電子情報通信学会 情報ネットワーク研究会, IN2004-99,Nov. 2004

- “アドホックネットワークにおけるオンデマンド型マルチパスルーティングプロトコル実装”
  電子情報通信学会 秋季全国大会, B-7-48, Sep. 2005

- “アドホックネットワークにおけるライブビデオ配信のための
  マルチパスルーティングプロトコル実装”
  電子情報通信学会 アドホックネットワーク時限研究専門委員会
  第 3 回ワークショップ, Jan. 2006

- “Experimental Evaluation of an On-demand Multipath Routing Protocol for Video Transmission in Mobile Ad hoc Networks”
  15th International Packet Video Workshop, Apr. 2006 (予定)

アドホックネットワークにおけるディスジョイント・マルチパスルーティング
Disjoint Multipath Routing for Mobile Ad Hoc Networks

谷山　健太　Kenta TANIYAMA
櫻井　祐介　Yusuke SAKURAI
甲藤　二郎　Jiro KATTO

早稲田大学大学院理工学研究科
Graduate School of Science and Engineering, Waseda University

1. はじめに

アドホックネットワークの研究の進展に伴い，さまざまなマルチパスルーティングプロトコルが研究されてきた．その中でも，経路の切断に対する耐性の向上や，負荷の分散に利用される，ディスジョイントなマルチパス（複数経路同士が互いに重複していない経路）を作るということがテーマになっているものが多く見られる．

本稿では，ソースルーティングの手法を AODV[1]に適用し，ノードディスジョイントなマルチパスを作成する手法を提案する．

2. 従来のマルチパスルーティング

AOMDV[2]では, RREQ に追加拡張した, Source ノード(以下 S ノード)から 1 ホップ目のノードを示すフィールドを用いて, S ノードと Destination ノード(以下 D ノード)間にリンクディスジョイントなパスを作成する．しかし，この手法では，S と D 間にある中間ノードが 2 つ以上共通する場合にディスジョイントパスを作成できない[3]．また，このことから中間ノードでジョイントしているパスを複数保持し，その中から利用可能なリンクを選ぶといったことになり，バックアップとして有効でない経路も多く作成し，エンド間遅延の増大にもつながる．

3. 提案手法

そこで，無効経路が少なくなるようなディスジョイントなパスの作成法について提案する．

まず，RREQ に対する拡張を行う．リクエストテーブル・RREQ パケットを拡張してソースルート情報を保持し，以下の手順でテーブルを更新していく．**1)**テーブルの RREQ シーケンス番号がパケットの RREQ シーケンス番号と等しく，かつテーブル上での，送信元 S からそのノードまでのホップ数よりもパケット上の送信元からそのノードまでのホップ数が同数以下ならば，テーブル上に第 2 経路として保持する．**2)**そして，RREQ を廃棄．それに当てはまらない場合は，AODV と同じ動作により，RREQ をそのまま廃棄する．

次に，RREP に対する拡張を行う．RREP にはソースルート情報とルートの分岐点を示すフラグを載せる．これにより，RREP の転送方向を第 2 経路に切り替え，ディスジョイントなパスを作成する．手順は以下の通り．**1)**D ノードは最初に届いた RREQ に対し，RREP を返す．**2)**D ノードは別経路から来た RREQ に対して, 同数ホップ以下であれば RREP の返信を開始する．**3)**その際，最初に届いた RREQ と，その最短ホップの RREQ のソースルートのアドレスを上流(S ノード側)から順に比較．両者が違う場合には，パスの分岐点とわかる．分岐後のノードを分岐フラグとして RREP に追加する．**4)**フラグのたったノードより下流(D ノード側)では，リクエストテーブルを参照し，予備ルートがあり，かつその予備ルートが分岐ノードよりも上流で合流している場合に，予備ルートに RREP の転送先を変える．**5)**もし別ルートがない，もしくは，上記の条件に当てはまらなかった場合には RREP をそのまま返す．

4. シミュレーション評価

ns-2 を用いたシミュレーションによって，AODV と，AODV をマルチパスに拡張した AOMDV，提案方式の比較を行った．シミュレーション条件は，エリア 2200[m]×600[m]，ノード数 100，S ノード 10，移動速度 0-20[m/s]として，UDP，512[bytes]，4[packets/s]でパケットを流し，300[sec]シミュレーションを行った．

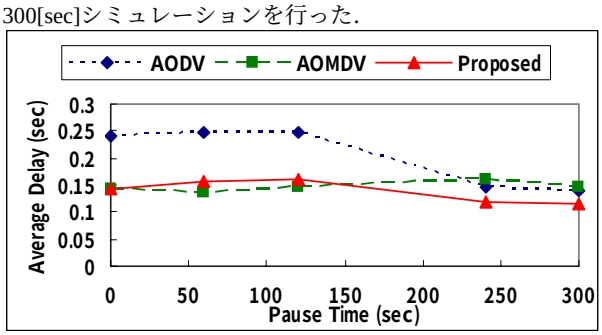


図１．　エンド間平均遅延時間

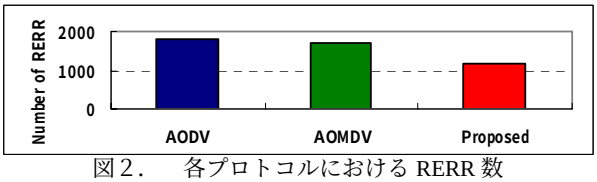


図２．　各プロトコルにおける RERR 数

5. まとめ

本稿において，提案方式の比較検討を行い，無効経路数を減らし，また，従来のマルチパスルーティングと同等以上の性能を示すことができた．

参考文献

[1] C. E. Perkins, et al., “Ad hoc On-demand Routing Vector (AODV) Routing”, RFC3561, July 2003
[2] M. K. Marina, S. R. DAS, “On-demand Multipath Distance Vector Routing in Ad Hoc Networks”, IEEE ICNP 2001
[3] 茂木信二 他, “アドホックネットワークのためのマルチパス・ルーチングの提案”, 信学技報, IN2002-125, 2002

ソースルートリストを用いたAODVマルチパス拡張

谷山 健太† 櫻井 祐介† 甲藤 二郎†

†早稲田大学大学院理工学研究科 〒169-8555 東京都新宿区大久保 3-4-1
E-mail: †{tani_ken, sakurai, katto}@katto.comm.waseda.ac.jp

あらまし 本稿ではオンデマンド型のマルチパスルーティングプロトコルを提案する．AODVは広く研究が行われているシングルパスルーティングプロトコルであるが，その経路再構築遅延を減らすために，様々なマルチパス拡張プロトコルが提案されている．我々はソースルートリストを用いることで，ホップ数の制約を受けずに不要なルーティングパケットを減らして経路を構築することができることを示す．ns-2によるシミュレーションでは他のプロトコルと比較して提案方式の有効性を明らかにすると共に，効果的な経路の構築方法を検証する．

キーワード アドホックネットワーク，マルチパスルーティング，AODV，ソースルート

AODV Multipath Extension using Source Route Lists

Kenta TANIYAMA† Yusuke SAKURAI† Jiro KATTO†

†Graduate School of Science and Engineering, Waseda University

3-4-1 Ohkubo, Shijuku-ku, Tokyo, 169-8555 Japan
E-mail: †{tani_ken, sakurai, katto}@katto.comm.waseda.ac.jp

Abstract This paper proposes an on-demand multipath routing protocol for mobile ad hoc networks. AODV is the widely studied single path routing protocol. A number of its multipath extension protocols have been proposed to reduce route discovery latency. We demonstrate that the route establishment using source route lists can reduce unnecessary routing packets without limitation of the hop count based approaches. We evaluate this proposed method comparing with the other protocols by using a simulator – ns2 and verify the efficient route establishment.

Keyword Mobile Ad Hoc Network, Multipath Routing, AODV, Source Route

1. はじめに

アドホックネットワークは，無線ノードによってマルチホップリンクを構築する技術であり，基地局といったインフラストラクチャを必要としない自律的なネットワークを構成する．その利用目的としては，災害発生時の緊急通信，イベント会場のような場所での一時的なネットワーク構成はもとより，無線技術の発達に伴い，ホームネットワークやセンサネットワークへの応用も考えられている．

アドホックネットワークの特徴として，ノードの移動や無線の衝突によりノード間のリンクブレークが頻繁に起こり，ネットワークトポロジが著しく変化することが挙げられる．また，無線機器は帯域や電力において有線機器よりも制限が厳しく，この制約の中でトポロジ変化に柔軟に対応しなければならない．このため，様々なルーティングプロトコルが提案・RFC化[1-4]されており，ルーティングの際に必要となる経路制御パケット数の抑制のために，データパケット通信時にのみ経路の作成と維持を行うオンデマンド型のルーティングプロトコルの研究が盛んに行われている．オンデマンド型の代表的なルーティングプロトコルである AODV[1]はルーティングのオーバーヘッドや経路の信頼性において優れたプロトコルであるが，オンデマンド型であるがゆえに，経路が切断された際に行われる経路探索の頻度とその経路探索に必要なメッセージ数の削減，また,経路切断からの素早い復帰といったことが問題となっている．

このため，シングルパスを構築するAODVをマルチパス構築型へ拡張する提案がなされている[5][6]．これらの研究では複数の重複しない経路（ディスジョイン

トパス）を構築することによって，一つのルートが無効になった際に他の予備ルートによって即座に通信を回復できることを目指している．しかし，AODV がホップ・バイ・ホップの通信であることからホップ数をベースに拡張をしているために，種々の制約を設けてしまい，柔軟な経路の選択ができない[7].

そこで，本稿では AODV の経路構築において，他のオンデマンド型のルーティングプロトコルである DSR[2]において使われているソースルートの概念を適用することで様々に経路を構築し，また，それらと従来のマルチパス提案とをシミュレーションにより比較評価することで，本提案の有効性とマルチパスにおける経路の効果的な選択方法を検証する．

2. 従来のマルチパスルーティング研究

ここではオンデマンド型のマルチパスルーティングプロトコルの経路探索に主眼を置いて，従来研究について述べる．なお，ディスジョイントパスには大別して 2 種類の考え方があり，一方をノードディスジョイント，他方をリンクディスジョイントと呼ぶ．ノードディスジョイントとは図 2.1 における経路 a と b, 経路 c と d のようにノード間で全く共有のない関係の経路を指し, リンクディスジョイントとは経路 a-c と b-d, あるいは a-d と b-c のような中間ノード 1 点を共有するような経路を指す．

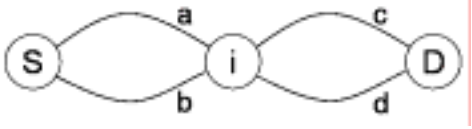


図 2.1　ディスジョイント概念図
ノードディスジョイント：a と b, c と d
リンクディスジョイント：a-c と b-d, or a-d と b-c

AOMDV[5]は RREQ(Route Requset)を拡張し, firsthop という概念を用いて，リンクディスジョイントなマルチパスを作成するプロトコルである．中間ノードが RREQ パケットを複数受信した場合に，advertised hopcount と呼ばれる S からそのノードへの経路表上の最大ホップ数と，パケットのホップ数を比較することで，経路のループを防ぐ．RREQ は firsthop フィールドに S から 1 ホップ目に通ったノードを記録し，ある中間ノードで重複受信された際に経路表上の hopcount list と比較，2 つの値が違えば，その中間ノードまでディスジョイントな経路が作成されたとして帰還経路を構築する. D は RREP(Route Reply)を複数返すことでリンクディスジョイントなマルチパスを S-D 間に作成する．AOMDV では firsthop の制約によって，重複経路を選んでしまう問題がある．

AODVM[6]は上記で述べた問題点に対処したマルチパスルーティングプロトコルであり, firsthop の代わりに jointcount の概念を用いてマルチパスを構築する．中間ノードが複数の帰還経路を保持している場合に，RREP 上の jointcount を増やし，帰還経路全てに RREP を送信する．その後，S において jointcount の値が小さい転送経路を選択することにより，ディスジョイントなパスを S-D 間に作成する．また keep-alive パケットによってバックアップ経路の品質を保証する．

しかし，jointcount による複数経路作成では S-D の経路が外に膨らんでいく形になるためネットワーク資源の消費に偏りが出るほか，ホップ数による制限のために，パケット送信に効率の悪いルートを選んでしまう可能性もある．

また，ソースルートを用いたオンデマンド型ルーティングプロトコルである DSR[2]をマルチパス型に拡張した Multipath DSR[8]では，RREQ を D が受信した際に複数の中から規則に従って経路を選択することによってマルチパスを構築する．一つ目がノードディスジョイントパス，二つ目が中間ノードから D へ向かって枝分かれした代替パスといった二つの規則が考案され，後者が良いパフォーマンスを示している．

本研究ではソースルートリストを用いているため，DSR のマルチパス拡張と相似する部分もあるが，相違点としては，ソースノードが最終的に経路を決めるという点やデータパケット送信時にはホップ・バイ・ホップで送信を行うという点が考えられる．

3. 提案手法

3.1. AODV 動作概要

AODV の動作としては大きく分けて 2 つのフェーズがあり，Route Discovery フェーズと Route Maintenance フェーズである．前者には RREQ・RREP パケット，後者には RERR(Route Error)パケットを用いる．

以下，動作概要を説明する．Route Discovery(図 3.1)では，ある送信元ノード S が宛先ノード D に対し通信を始めたいときに，S は RREQ をブロードキャストする．中間ノードにおいて RREQ を受信した場合，最初に届いた RREQ に関して中間ノードは，S への帰還経路となる次ホップとシーケンス番号を経路表に記録し RREQ を再ブロードキャストする．もし RREQ パケットを複数受信した場合，そのシーケンス番号が同数以下, つまり古い RREQ であった場合に即座に廃棄する．D が RREQ を受信すると，RREP を S へ向けてユニキャスト送信する．中間ノードにおいて RREP を受信すると，そのノードでは D への転送経路となる次ホップとシーケンス番号を経路表に記録し，帰還経路上の隣接ノードへ RREP を転送する．RREP が S に着くと，S

から D へのデータ転送経路が確立される．Route Maintenance では，あるノードが定期的に通知される Hello パケットを受信できない，あるいは MAC 層から送信失敗の通知があったということによりリンクブレークを検知すると，RERR を送信する．S が RERR を受信すると，Route Discovery フェーズに戻り，再び経路の探索が行われる．

ノードの移動のような原因でリンクブレークが頻繁に起きる場合に，経路再探索のプロセスによるデータパケットの送信遅延の増加や制御パケットの増大により，パフォーマンスの低下を招く．

図 3.1　AODV の Route Discovery

3.2 AODV のマルチパス拡張

Route Discovery に際して，複数経路探索のために AODV に対して AOMDV では firsthop，AODVM では jointcount フィールドを用いていたが，本提案ではソースルートリストを RREQ･RREP パケットに適用する．そして，ソースルートリストを用いる際に，一定のメトリックに基づいて各ノードにおいてパケットの転送・廃棄を行う．

(1) RREQ 拡張

ノード S は通信を行う際に RREQ をブロードキャストする．ある中間ノードが初めて RREQ を始めて受信した場合，帰還経路を経路表に記録し，RREQ にはソースルート情報として自分のアドレスを RREQ に記録する．また，その中間ノードへの到着が重複した RREQ を受信した場合，ソースルートリストを確認してルーティングループが起きている場合には即座に RREQ を破棄する．そして，中間ノードではホップ数・遅延といったメトリックにより，遅れて到着した重複 RREQ を受け入れて帰還経路を更新するかどうかを決定する．重複した RREQ は経路表更新後に再ブロードキャストされない．一つの中間ノードが保持する帰還経路数は *MAX_PATH* 本に限られ，*MAX_PATH*=3 としている。中間ノードにおける経路表は図のようになり，ルート毎に情報が記録される．これを繰り返し，RREQ がノード D に到着する．

(2) RREP 拡張

ノード D が RREQ を受信すると RREP をノード S に向かってユニキャストする．最初に受信した RREQ に対しては RREP 生成を即座に行う．これはシングルパスの AODV で作られるルートとほぼ同じである．ノード D での重複 RREQ は中間ノードと同じく特定のメトリックで受け入れるかどうか決定される．

RREP パケットを受信した中間ノードは帰還経路に沿って RREP を転送するが，帰還経路が複数ある場合にはバイキャスト(複数ユニキャスト)により送信する到着が遅れた重複 RREP は特定のメトリックに応じて転送経路を更新するかが決定される．転送経路の保持数 *MAX_PATH'*は，帰還経路と同様 *MAX_PATH'*=3 本としている．以下図 3.2 において RREQ と RREP の重複受信時における中間ノードの転送動作をまとめる．

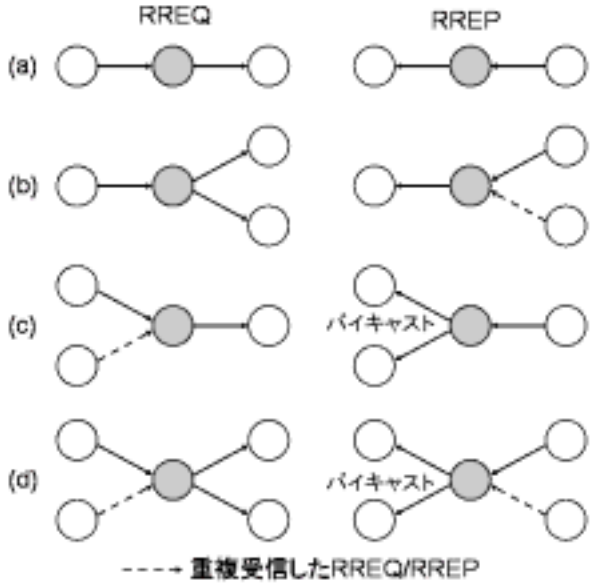


図 3.2　RREQ と RREP の転送動作. (a)シングルパス形成，(b)複数転送経路の更新(重複 RREP による)，(c)複数帰還経路の更新(重複 RREQ による)，(d)複数帰還・転送経路の更新(重複 RREQ と重複 RREP による)

図 3.3　提案手法における Route Discovery

RREP がノード S に到着すると，最初に到着した RREP によって構築された経路を第 1 経路とし,即座にデータパケットの転送を開始する．遅れて到着した複数の重複 RREP にて構築された経路を一定のメトリックで受け入れたものをバックアップ経路として経路表に保持する．

(3) 経路選択メトリック

経路選択メトリックにおける経路作成法として三つの手法を考える．一つ目がホップ数に基づくもの(手法Ⅰ)，二つ目が遅延に基づくもの(手法Ⅱ)，そして三つ目がソースリストによるディスジョイント検出に基づくもの(手法Ⅲ)である．

手法Ⅰでは帰還経路・転送経路ともに重複 RREQ/RREP が最小ホップ数であったときに経路を更新する．そして，重複 RREP は即座に廃棄される．ノード S は保持する経路のうち最短のものを利用し，バックアップ経路に関しても最短のものに更新する．手法Ⅱでは RREQ/RREP が早く到着したものから経路を更新する．重複 RREP は手法Ⅰと同様に即座に廃棄される．ここで，経路更新に関わる重複 RREP のうち，早く到着してもホップ数が以前のものよりホップ数が大きければ，バックアップ経路として採用しない．これは，より遅延が少ない経路でもホップ数が大きければリンク切断の可能性が増加するといった研究背景[8]に基づく．手法Ⅲでは RREQ, RREP の転送を経路の分岐情報を基にして決定し，帰還経路・転送経路の経路更新を行う．この手法を図 3.3 で示す．

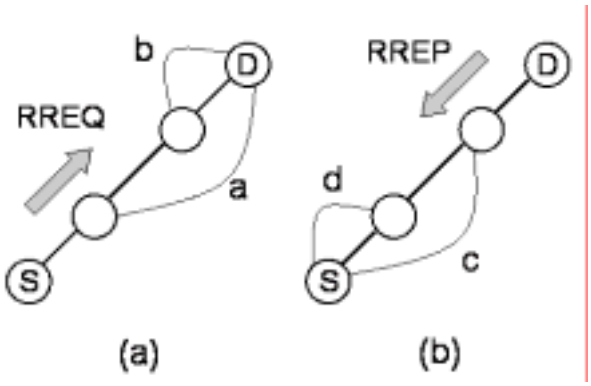


図 3.4　ソースルート情報による経路更新

図 3.4 において直線部で示されるのが最初に構築される経路(以下，プライマリルート)であるとする．(a)において a と b が代替経路の候補であるとし，S-D 間のホップ数は同数とする．ホップ数は同数であっても，a は b よりもプライマリルートに対し重複経路が少なく，リンク切断に対し強い．こういった a のような経路をノード D で判断し，RREP を返送することができる．また，(b)においても同様にノード S は，代替経路として，プライマリルートに対して重複経路の少ない c を選ぶことができる，このような考え方で，代替経路 b や d を選ぶことなく両方向経路で最大限にディスジョイントな経路を選択する．これは AOMDV や AODVM といったホップ数をもとにしたプロトコルでは実現できない．

4. シミュレーション評価

4.1 シミュレーション条件

本提案を，ns-2[9]を用いたシミュレーション実験により評価を行った．比較対象はシングルパスルーティングである AODV，マルチパスルーティングである AOMDV と本提案の手法Ⅰ，Ⅱ，Ⅲである．実験 1 でのシミュレーション条件は表 4.1 に示すとおりである．実験 1 では端末の移動速度を変え，各ルーティングプロトコルの特性を見る．実験 2 でのシミュレーション条件は表 4.2 に示すとおりである．実験 2 では最大移動速度 5[m/s]といった急でない端末移動速度でストリーミング通信をしたような場合を想定している．なお，表 4.1 にて記載され，表 4.2 に記載されていない部分は同条件としている．

シミュレーション時間	500[sec]
移動範囲	2200×600[m]
端末数	100
通信端末数	10 組
移動モデル	Random Way Point
MAC プロトコル	IEEE802.11
リンク速度	2[Mbps]
通信範囲	250[m]
トランスポートプロトコル	UDP
送信レート	16[kbps] (512[bytes],4[packets/s])

表 4.1　実験 1 のシミュレーション条件

シミュレーション時間	300[sec]
通信端末数	2 組
送信レート	128[kbps] (1024[bytes],16[packets/s])

表 4.2　実験 2 のシミュレーション条件

4.2 シミュレーション結果

(1) 実験 1

図 4.1 にエンド間平均遅延時間，図 4.2 にパケット到着率，図 4.3 にルーティングパケット数を示す．図 4.3 では RREQ/RREP/RERR の 3 つのルーティングパケ

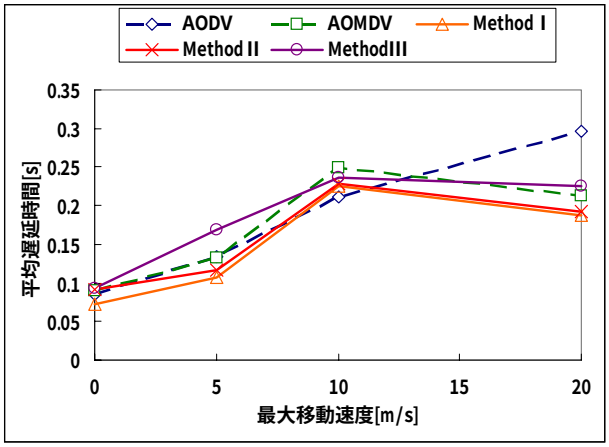


図 4.1　エンド間平均遅延時間

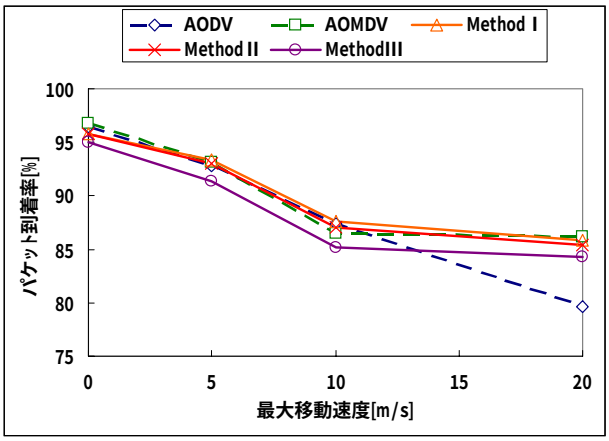


図 4.2　パケット到着率

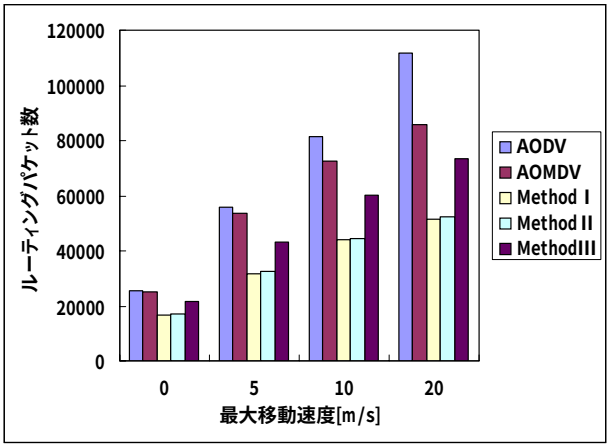


図 4.3　ルーティングパケット数

図 4.1 においてエンド間の平均遅延時間は，最大移動速度が低いときには，AODV，AOMDV，提案との差はあまりないが，移動度が高くなると，AODV のパフォーマンスは下がっていく．これは，ノードの移動によって経路が切れやすくなる場合に各マルチパスルーティングプロトコルと違いバックアップ経路を保持していないことから経路探索遅延が増えることに起因する．また，概して提案手法Ⅰ，手法Ⅱ，AOMDV の順に遅延時間が少ない．手法Ⅰでは経路選択を遅延優先にしているので遅延が短くなる．AOMDV では RREP が中間経路でバイキャストされない．このため経路保持数は提案手法よりも減るため，経路探索が増え，遅延の原因となる．

図 4.2 においてパケット到着率も，エンド間遅延と同様，移動度が高くなると AODV のパフォーマンスが落ちていく．マルチパスルーティングプロトコルでは手法Ⅰの到着率が若干高いが，これは 3.2(3)で述べたように，短いホップ数ではリンク切断の可能性が少なくなるためである．

図 4.3 と図 4.1 を比べると，ルーティングパケットが少ない方が，遅延が少なくなる傾向が読み取られる．これは，パケット衝突回避のため，パケット数が増えると送信遅延時間が発生すると考えられる．ただ，AOMDV と提案手法Ⅲを見て分かるように，必ずしも遅延と傾向が同じというわけではない．なぜなら，負荷が少ない状況(実験 1 では 16kbps×10 コネクション程度)では Route Discovery でコネクションを張りなおすよりも，始めにたくさんのルートを保持して，無効になったリンクのみ RERR で削除していく方(リンクディスジョイント型のアプローチ)が，経路が最適化されるからである．

(2) 実験 2

図 4.4 に実験結果を示す．マルチパスプロトコル群のうちリンクディスジョイント型のアプローチは軒並み遅延時間が延びている．提案手法Ⅲは AODV よりも遅延時間，到着率ともに若干良くなっている．

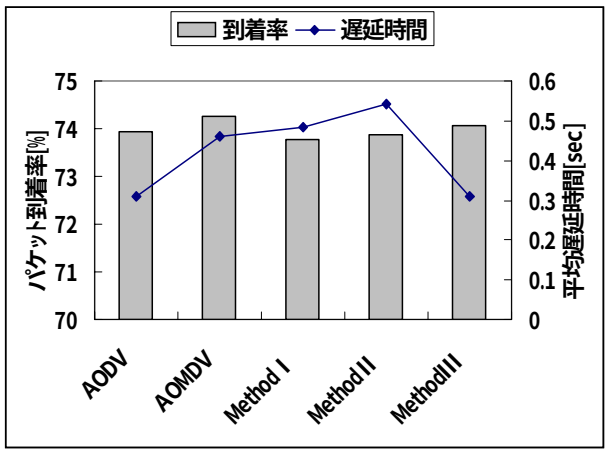


図 4.4　パケット到着率とエンド間平均遅延時間

(1)の最後に述べたが，先ほどと比べてネットワークにはより負荷がかかった状態であり，このような状況下ではバックアップ経路はノードディスジョイント型の方が優れた特性を示している．

5. まとめ

本稿では，ソースルートリストを用いたアドホックネットワークのマルチパスルーティングを提案し，AODV を拡張した．マルチパスルーティングの有効性を示し，また，どういったメトリックで経路を構築するのが効果的かを比較するため３手法をシミュレーション評価した．

今後，メトリックに関する更なる検証を進めるとともに，今回シミュレーションによってある程度の有効性が確認された事項を実装へ反映することが課題である．

文　献

[1]C.Perkins, E.B-Royer, S.Das, “Ad hoc On-demand Routing Vector (AODV) Routing”, RFC3561, July 2003.

[2]D.Johnson, D.Maltz, Y-C.Hu, “The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks,” Internet Draft, draft-ietf-manet-dsr-10.txt, July 2004, Expired.

[3]T.Clausen, P.Jacquet, “Optimized Link State Routing Protocol (OLSR),” RFC3626, Oct. 2003.

[4]R.Ogier, F.Templin, M.Lewis, "Topology Dissemination Based on Reverse-Path Forwarding (TBRPF)," RFC3684 , Feb. 2004.

[5]M.Marina, S.Das, “On-demand Multipath Distance Vector Routing in Ad Hoc Networks”, IEEE ICNP, 2001

[6]茂木信二，吉原貴仁，堀内浩規，“アドホックネットワークのためのマルチパス・ルーチングの提案，”信学技報, IN2002-125, pp.51-56, Nov. 2002.

[7]Yusuke Sakurai, Jiro Katto, “AODV Multipath Extension using Source Route Lists with Optimized Route Establishment,” Inter National Workshop on Wireless Ad-hoc Networks(IWWAN), May. 2004.

[8]A.Nasipuri, R.Castaneda, S.Das, ”Performance of Multipath Routing for On-Demand Protocols in Mobile Ad Hoc Networks,” Mobile Networks and Applications, vol6, 2001.

[9] “ns-2” http://www.isi.edu/nsnam/ns/

アドホックネットワークにおける オンデマンド型マルチパスルーティングプロトコル実装

An Implementation of On-demand Multipath Routing Protocol for Mobile Ad Hoc Network

森井 健之 Takeshi Morii　谷山 健太 Kenta Taniyama　甲藤 二郎 Jiro Katto

早稲田大学大学院理工学研究科
Graduate School of Science and Engineering, Waseda University

１． はじめに

アドホックネットワークは，無線ノードによってマルチホップリンクを構築する技術であり，基地局といったインフラストラクチャを必要としない自律的なネットワークを構成する．

アドホックネットワークの特徴であるネットワークトポロジの著しい変化に対応するために様々なルーティングプロトコルが提案され，マルチパス構築型へ拡張する提案がなされている[1]．しかし，多くのマルチパス型のプロトコルの提案に関する評価は，シミュレーションでの有効性は挙げられているものの，アプリケーションなどと絡めた実装評価による有効性は示されていないことが多い．そこで，本研究ではシングルパスを形成するオンデマンド型のルーティングプロトコルである AODV をマルチパス型に拡張したものを実装することで，動的にマルチパスを形成した際のパフォーマンスを評価する．

2 ．提案方式

本研究では NIST による AODV の実装コードである KernelAODV[2]を利用し，Linux 上へのマルチパスルーティングプロトコルの実装を行う．KernelAODV の動作をマルチパス型に変えるに当たって，以下のような拡張を行う．パラメータの設定に関しては[1]を参照にした．

(1)RREQ 拡張

送信元 S は通信を行う際に RREQ をブロードキャストする．ある中間ノードが初めて RREQ を受信した場合，帰還経路をルーティングテーブルに記録し，RREQ にはソースルート情報として自分のアドレスを RREQ に記録する．また，その中間ノードへの到着が重複した RREQ を受信した際，ソースルートリストにより経路にループが確認された場合には即座に RREQ を破棄する．そして，中間ノードではホップ数・パケット到着遅延といったメトリックにより，遅れて到着した重複 RREQ を受け入れて帰還経路を更新するかどうかを決定する．一つの中間ノードが保持する帰還経路数は MAX_PATH 本に限られ，本稿の実験では MAX_PATH=3 とする．

(2)RREP 拡張

宛先 D が RREQ を受信すると RREP をノード S に向かってユニキャストする．最初に受信した RREQ に対しては RREP 生成を即座に行う．これはシングルパスの AODV で作られるルートとほぼ同じである．重複受信した RREQ は中間ノードと同じく特定のメトリックで受け入れるかどうか決定される. RREP パケットを受信した中間ノードは帰還経路に沿って RREP を転送するが，帰還経路が複数ある場合にはバイキャスト(複数ユニキャスト)により送信する. 到着が遅れた重複 RREP は特定のメトリックに応じて，転送経路を更新するかが決定される．転送経路保持数 MAX_PATH' は，帰還経路と同様 3 本とする．

なお，図 1 には AODV と，ソースルートリストを加えた提案方式のルーティングテーブルを示す．また図 2 には，RREQ 受信時の経路更新メトリックを示す．

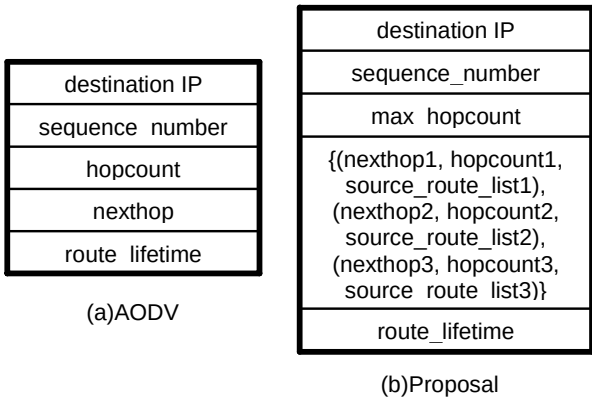


図1. AODVと提案方式のルーティングテーブル

if ($seqnum_i^d < seqnum_j^d$) **then**
　古いルートを破棄
　新規ルーティングエントリ作成(第一経路作成)
else if ($seqnum_i^d = seqnum_j^d$)
　and ($\max\_ hopcount^d > hopcount$
　and
($number\_of\_routes < MAX\_PATH$) **then**
　代替経路を挿入(第二，第三経路作成)
endif

図2. 経路更新メトリック

3．実装評価

本提案を KernelAODV 上に組み込み, Linux マシン数台を用いてパフォーマンスを測定し， 発表時に紹介する．

4．おわりに

オンデマンド型マルチパスルーティングプロトコルの提案及び実装について述べた．今後はアプリケーションの特性を考慮し，電波強度やデータ送信レート制御による経路品質管理を行う予定である．

参考文献

[1]谷山健太 他, “ソースルートリストを用いた AODV マルチパス拡張”, 信学技報, IN2004-99, 2004

アドホックネットワークにおけるライブビデオ配信のための マルチパスルーティングプロトコル実装

谷山 健太† 森井 健之† 小泉 信也† 小谷 幸宏‡ 野口 和浩‡ 甲藤 二郎†‡

†早稲田大学大学院理工学研究科 〒169-8555 東京都新宿区大久保 3-4-1
‡早稲田大学理工学部 〒169-8555 東京都新宿区大久保 3-4-1
E-mail: {tani_ken, morii, shinya, kotani, kazuhilo, katto}@katto.comm.waseda.ac.jp

あらまし 本稿ではビデオ配信を目的としたオンデマンド型のマルチパスルーティングプロトコルを提案・実装し, 実世界での性能評価を行う. 現在広く知られているシングルパスルーティングプロトコルである AODV は, 様々なマルチパスルーティングプロトコルに拡張されている．我々はソースルートリストを用いた提案手法[4]により, ホップ数の制約を受けずに経路探索遅延を減らし，安定ルートを自動的に選び，より効果的にライブビデオ配信をすることが出来ることを示す．我々は提案方式と AODV について比較するため，８台のノート PC にルーティングプロトコルを実装し，ライブストリーミング実験を行った.

キーワード アドホックネットワーク，AODV，マルチパスルーティング，ソースルート

Implementation of an On-demand Multipath Routing Protocol for Live Video Transmission in Mobile Ad hoc Networks

Kenta TANIYAMA† Takeshi MORII† Shinya KOIZUMI†

Yukihiro KOTANI‡ Kazuhiro NOGUCHI‡ Jiro KATTO†‡

†Graduate School of Science and Engineering, Waseda University 3-4-1 Ohkubo, Shijuku-ku, Tokyo, 169-8555 Japan

‡Science and Engineering, Waseda University 3-4-1 Ohkubo, Shijuku-ku, Tokyo, 169-8555 Japan
E-mail: {tani_ken, morii, shinya, kotani, kazuhilo, katto}@katto.comm.waseda.ac.jp

Abstract We propose an on-demand multipath routing algorithm in a mobile ad hoc network for video transmission and evaluate its real world performance of video streaming application. There have been a number of multipath routing protocols extended from AODV which is a well-known single path routing protocol. Multipath routing protocols indicate good performance in the reduction of route discovery latency and unnecessary routing packets in simulations. We show that the route establishment using source route lists provided by us [4] can reduce the route discovery latency, select stable routes automatically, and work well for live video streaming without limitation of the hop count based approaches. We evaluate this proposed method comparing with the original AODV by using eight laptop PCs and demonstrate live streaming experiments.

Keyword Mobile Ad hoc Networks, AODV, Multipath Routing, Source Routes

1. はじめに

アドホックネットワークでは，基地局やアクセスポイントといったインフラストラクチャを必要とせずに安価で容易にネットワークを構築することが出来る．各ノードはマルチホップの無線通信によってエンド間の通信路を確保することが出来るが，ノードの移動や不安定な電波特性により，通信路中のリンクブレークが頻繁に起こりやすく,ネットワークトポロジが変化しやすい．このため，IETF では，様々なルーティングプロトコルを RFC 化しており，現在，PMP (Proactive MANET Protocol) と RMP (Reactive MANET Protocol) という２つのタイプのルーティングプロトコルが議論されている. RMP として標準化の進む DYMO は AODV (Ad hoc On-demand Distance Vector)[1]が基になっている．AODV は低ルーティングオーバーヘッドやトポロジ変化に強いといった特長を持つが，シングルパス型のルーティングプロトコルなので，リンクブレークが起きるとルートを再構築する必要がある．このため，経路再探索による遅延やパケットロスが起き，ライブストリーミングのような信頼性のある通信アプリケーションにとって深刻な問題となる.

また，経路再探索遅延を低減するために，オンデマ

ンド型のマルチパスルーティングプロトコルの提案がなされ，シミュレーション評価されてきた[2]-[5]．これらの研究ではデータ送信ルートと重複しない(ディスジョイントな)バックアップルートを保持することで通信の素早い回復を図っている．結果，マルチパスルーティングプロトコルはシングルルーティングプロトコルよりも良い性能を示しているが，実世界でのマルチパスルーティングプロトコルの使用に対する評価としては十分ではないと考える．

そこで本研究では AODV をマルチパス型に拡張してノート PC に実装し，性能評価並びにライブストリーミングアプリケーションによる配信を行った．我々の提案では RREQ/RREP をソースルートリストを用いて拡張することにより，ループのない経路を構築するとともに，遅延やホップ，ディスジョイント性に応じて中間経路を決めることが可能である．

2. 提案プロトコル概要

本研究では NIST による AODV の実装コードである KernelAODV[7]を利用し, Linux 上へのマルチパスルーティングプロトコルの実装を行う．KernelAODV の動作をマルチパス型に変えるに当たって，以下のような拡張を行う．

2.1. Route Discovery 拡張

1)RREQ 処理

送信元 S は通信を行う際に RREQ をブロードキャストする．ある中間ノードが初めて RREQ を受信した場合, 帰還経路をルーティングテーブルに記録し, RREQ には自分のアドレスをソースルート情報として記録する．その中間ノードが，別の近隣ノードから遅れて同一 ID の RREQ (Delayed RREQ)を受信した場合，中間ノードはソースルートリストを確認する．そのノードの IP アドレスが含まれていればループが起きたと判断し，即座に重複 RREQ を破棄する．そのノードの IP アドレスが含まれておらず，ホップ数が最初に受信された RREQ のルートより大きくなければ，到着順にバックアップの帰還経路としてルーティングテーブルの nexthop として IP アドレスを保持した後，破棄する．このルーティングパケットの経路構築メトリックを”delay metric”[4]と呼ぶこととする．また，RREQ によって発見された経路は HELLO によって構築された 1 ホップのルートよりも高い優先度をもつこととしている．一つの中間ノードが保持する帰還経路数は MAX_PATH 本に限られ，本稿の実験では MAX_PATH=3 とする．

図 1 においてルーティングテーブルの拡張を示す．nexthop, hopcount, route lifetime は帰還経路毎に保持される．

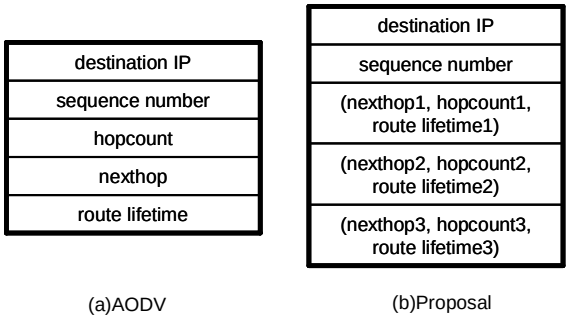


図 1．提案手法によるルーティングテーブル拡張

2)RREP 処理

あて先 D が RREQ を受信すると RREP をノード S に向かってユニキャストする．最初に受信した RREQ に対しては RREP 生成を即座に行う．これはシングルパスの AODV で作られるルートとほぼ同じである．ノード D で Delayed RREQ が受け入れられた場合は，バックアップルートに対して RREP を送信する．RREP パケットを受信した中間ノードは帰還経路に沿って RREP を転送するが，帰還経路が複数ある場合には[3]に見られるような複数ユニキャストにより送信する．到着が遅れた重複 RREP (Delayed RREP)は，RREQ 同様 delay metric によってルーティングテーブルのバックアップ転送経路を到着順に更新する．転送経路の最大保持数 MAX_PATH'は, 帰還経路と同様 3 本とする．データの転送は最初のルートが確立された時点から始まる．図 2 に経路構築の例を示す．

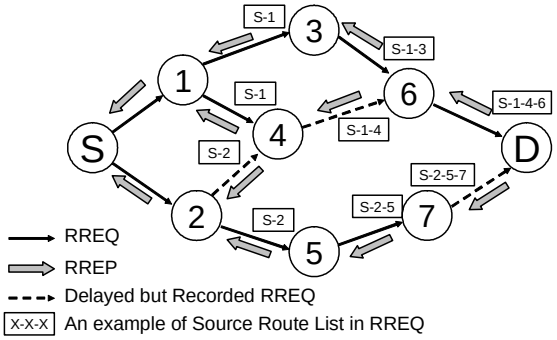


図 2．提案による Route Discovery 拡張

2.2. Route Maintenance 拡張

1)RERR 処理

ノードが HELLO メッセージを近隣ノードから受信できなかった場合，ノードはリンクブレークを検知する．もしバックアップルートがないならば，ルーティングルーティングテーブル内のルートを無効にして RERR をプリカーソルリスト上のノードに送信する．バックアップルートがあるならば中間ノードにおいて送信ルートを切り替える．経路再探索をしないことに

より，パケットの遅延やルーティングパケットの送信量を減らすことが出来る．また，HELLO パケットはリンクブレーク検知だけでなく，バックアップルートのタイマを更新し，ルートの有効期限を延長させることができる．

3. 提案プロトコル評価

3.1. プロトコルテストベッド

実装環境として，我々は IBM ThinkPad 8 台 (Celeron 700MHz，メモリ 256MB×4 台，Celeron 800MHz，メモリ 128MB×4 台)，OS には Red Hat Linux 9(kernel version 2.4.20)を用いた．無線 LAN インターフェースとしては IEEE802.11b 準拠カードを oricono_cs ドライバ上で 動作させた．無線 LAN の設定としては，同一チャネル，RTS/CTS 無し，送信レートは，実装上の問題点としてよく知られるグレーゾーン問題[6]を回避するために，2Mbps 固定とした．また,ノード同士の通信を可能にするために ad hoc demo モードで実験を行った．この ad hoc demo モードは IEEE802.11 標準の IBSS ad hoc モードとは異なる BSSID(cell)の同期を必要としない ad hoc モードであり，同一ルーティングプロトコルのネットワークを異なる BSSID のセグメントに分割するといった弊害を生むことがない．

3.2. 実装実験環境

我々は屋内において 2 種類の実装実験を行った．図３に示すように MAC フィルタリングによって意図的に作り出したトポロジの場合と，図４のようにノードを配置し，通信状況によってトポロジがある程度変化する環境下での場合においてそれぞれ評価する．

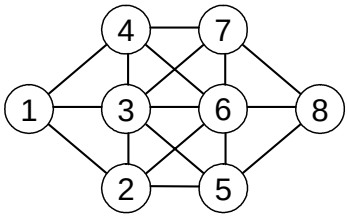


図３．MAC フィルタリングによるトポロジ

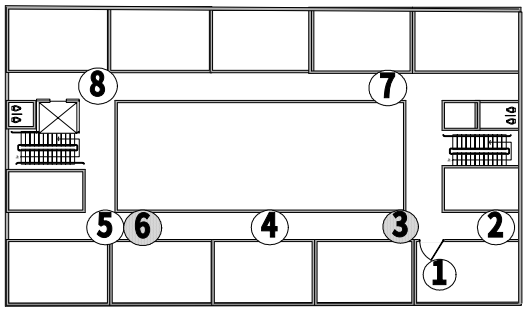


図４．ノード配置; ノード 3 と 6 は 3 階，その他のノードは 4 階に配置

3.3. 評価

1) 通信回復時間

Ping を用いて，経路が切断してから通信が回復するまでの平均時間を計測した．図３において，ノード 1 は 512byte，10packets/s という比較的穏やかなトラフィック負荷でノード 8 に対して通信を行う．この環境下で 通信中の中間ノードをランダムにダウンさせ,経路探索がどの程度影響を及ぼすかを観測した．図５にその結果を示す．提案手法では，ルーティングテーブルから送信ルートが削除された後バックアップ経路に切り替えるため，平均通信回復時間を短縮できていることが分かる．

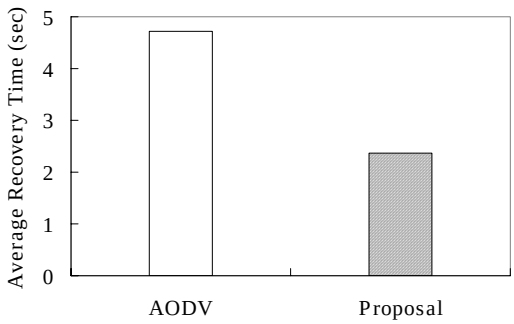


図５．平均通信回復時間

2) スループット比較

図４におけるノード 1 からノード 8 へ 30 秒間 TCP ストリームを流したときのスループットをこの実験では計測した．図６は平均スループット，図７はそのスループットの計測値の分布である．図６において 0kbps は経路探索がタイムアウトしてしまったことを示している．提案手法では平均 349kbps であり，AODV の約 1.2 倍の値を示している．

図７のスループット分布を見ると，提案手法と AODV の分布に相似性が見られる．これは，経路探索フェーズや HELLO 交換において，不安定ではあるが最小ホップであるというルートをたどりやすいことが原因であると考えられる．例えば，ノード 1 から 2,3 へのリンクは安定であるが，これらを経由するとノード 8 へは 3 ホップでたどり着き，ノード 1 から 4,5 へのリンクは不安定ではあるが，2 ホップでノード 8 へたどり着く．このためノード 1 での AODV は不安定な 4,5 を次ホップとして何度も選択し，何度も経路探索を引き起こしてしまう．我々の提案では HELLO メッセージは主にリンクのタイマ更新にしか使われないうえ,たとえ不安定リンクを選択してしまったとしても，バックアップ経路のリンクに置き換えることが出来るため，AODV ほどの影響は受けない．実際，提案手法では経路切断が起きると安定したルートに素早く収束

していたが，AODV は安定ルートが構築されるまで頻繁に経路探索を繰り返していた．

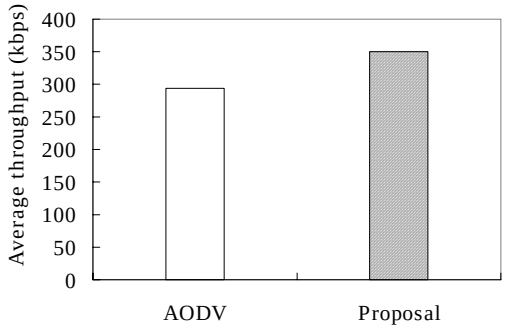


図６．平均スループット

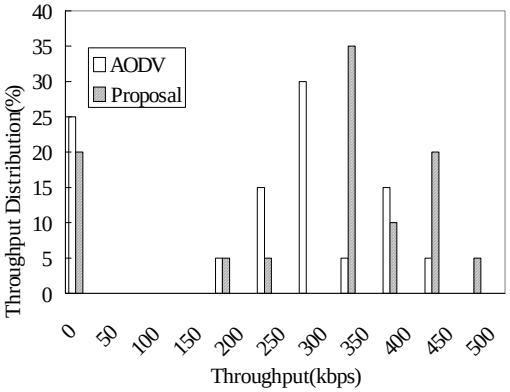


図７．スループット分布

3.4 ライブビデオ配信

図５におけるノード 8 に USB カメラである Logitech QuickCam Pro 4000 を取り付け，RealVideo をコーデックとして用いた．ビデオのビットレートは 128kbps, サイズは 320*240pixels, フレームレートは 15fps である．図８にデコードされた画像を示す．RealPlayer を起動させたノード 1 からノード 8 へストリーミングを要求すると，ノード 1 は経路探索を開始し，経路確立後にストリーミングの接続要求がノード 8 へ通ることとなる．

AODV の場合，2,3 秒後に接続要求とバッファリングが終了し，ビデオは問題なく再生され，画質に関しても大きなパケットロスは見受けられなかった．しかし，少しした後に接続が切れてしまい，タイムアウトエラーが頻発した．これは先に述べたように不安定なリンクを選んでしまいやすい AODV の性質によるものであると考えられる．

一方，提案手法でも AODV と同様ビデオはストレスなく再生された．また，パケットロスもほとんどなく，タイムアウトせずに長時間ビデオを見ることが出来た．

図８．デコードされたライブビデオ配信画像

4. まとめ

本稿ではマルチパスルーティングプロトコルをノート PC 上に実装し，性能を評価するとともにライブストリーミング配信のデモンストレーションを行い，従来の AODV の通信性能が改善されることを示した．我々の提案では，不安定なリンクを含むルートは通信中に徐々に安定なルートへ置き換えられ，かつ,バックアップ経路に切り替わる際にストリーミングビデオはタイムアウトすることなくシームレスに再生された．

今後は，パケットの受信電力(SNR)を用いての，ビデオ圧縮レイヤと協調した MAC 層における送信レート制御や,マルチパスへの複数ストリーム配信のため，MDC(Multiple Description Coding) [8]を用いたマルチチャネル送信などを検討していく予定である．

文　献

[1]C.Perkins, E.B-Royer, S.Das, “Ad hoc On-demand Routing Vector (AODV) Routing”, RFC3561, July 2003.

[2]M.Marina, S.Das, “On-demand Multipath Distance Vector Routing in Ad Hoc Networks”, IEEE ICNP, November 2001

[3]S. Motegi, H. Horiuchi, and Members, “AODV-Based Multipath Routing Protocol for Mobile Ad Hoc Networks,” IEICE Trans. Commun., Vol.E87-B, No.9 September 2004

[4]Y. Sakurai, J. Katto, “AODV Multipath Extension using Source Route Lists with Optimized Route Establishment,” International Workshop on Wireless Ad-hoc Networks (IWWAN), May 2004.

[5]A.Nasipuri, R.Castaneda, S.Das, “Performance of Multipath Routing for On-Demand Protocols in Mobile Ad Hoc Networks,” Mobile Networks and Applications, vol6, August 2001.

[6]H. Lundgren, E. Nordstrom, and C. Tschudin, “Coping with communication gray zones in IEEE 802.11b based ad hoc networks,” In ACM international workshop on Wireless mobile multimedia (WoWMoM), September 2002.

[7]Kernel AODV 'http://w3.antd.nist.gov/wctg/aodv_kernel/'

[8]V. K. Goyal. "Multiple Description Coding: Compression Meets the Network," IEEE Signal Processing Magazine, September 2001.