

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

茂木 信二[†] 吉原 貴仁[†] 堀内 浩規[†]

[†](株)KDDI 研究所 〒356-8502 埼玉県上福岡市大原 2-1-15

E-mail: [†]{motegi, yosshy, hr-horiuchi}@kddilabs.jp

あらまし アドホックネットワークのためのルーチングである AODV (Ad hoc On-demand Distance Vector Routing) は、経路発見と呼ばれる広報により経路の作成を行う。ノードの移動に伴い 1 つの経路が切断するたびに行う頻繁な経路発見を抑制するため、複数の経路を作成する拡張を行ったルーチングがこれまでに提案されている。しかしながら、作成する経路が循環してしまう問題やリンクが重複しない経路の作成を行うことができなくなってしまう問題があった。

そこで本稿では、宛先に至る経路が交差するノードに着目することで互いにリンクが重複しない経路の作成と維持を可能とするマルチパス・ルーチングをあらたに提案する。

キーワード アドホックネットワーク、マルチパス、リンクが重複しない複数経路

Proposal on Multipath Routing for Ad hoc Networks

Shinji MOTEGI[†] Kiyohito YOSHIHARA[†] and Hiroki HORIUCHI[†]

[†]KDDI R&D Laboratories Inc. 2-1-15 Ohara, Kamifukuoka-shi, SAITAMA, 356-8502 Japan

E-mail: [†]{motegi, yosshy, hr-horiuchi}@kddilabs.jp

Abstract A single path routing protocol for mobile ad hoc networks known as AODV (Ad Hoc On-demand Distance Vector) is widely studied. To reduce frequent route discovery, some multipath routing protocols, which extend AODV, have been proposed. However, these protocols have some issues; leading to routing loops between neighbor nodes and having a case when the protocol could not find disjoint routes, even if there exist the routes.

In this paper, we propose a new multipath routing protocol for ad hoc networks. The proposed protocol enables to find disjoint paths by directing our attention to nodes that have multiple reverse routes and provide multiple routes maintenance.

Keyword Ad Hoc Networks, Multi-path, Link disjoint-path

1. はじめに

アドホックネットワークは、ノードの移動に伴ってネットワーク構成が著しく変動する。そのようなネットワークにおいてルーチングに必要なオーバーヘッドを抑制するため、データパケットの転送に必要な経路のみの作成と維持を行うオンデマンド型のルーチング・プロトコルの研究が広く行われている。オンデマンド型のルーチング・プロトコルは、ネットワーク全体への広報を用いる経路発見と呼ばれる処理により経路を作成する。宛先が受信できるデータパケット数の増加と送信元から着信するまでに要する時間を短縮するためには、ノードの移動に伴い経路が切断されるたびに行われる頻繁な経路発見を抑制することにより広報するメッセージ数の削減と経路発見に要する時間の短縮を図ることが重要となる。

これまでにオンデマンド型のシングルパス・ルーチングとして知られる AODV (Ad hoc On-demand Distance Vector Routing) [1] を拡張し 1 回の経路発見で複数の経路を作成することで 1 つの経路が切断した時に残りの予備とする経路への切り替えを行うマルチパス・ルーチング・プロトコルが提案されている [2][3]。しかしながら、作成した経路が隣接ノード間で循環してしまう問題や 2 つのノード間の経路において互いに共通するリンクがないことから一般に経路の切断に対

する耐障害性を提供するリンクが重複しない経路を作成することができなくなってしまう問題があった。

そこで本稿では、オンデマンド型のマルチパス・ルーチングをあらたに提案する。提案するルーチングは AODV を拡張することで実現し、宛先に至る経路が交差するノードに着目することで互いにリンクが重複しない経路を作成する。作成した一方の経路をデータパケットの転送に用いる経路、残りの他方を予備の経路とし予備の経路を維持するための処理方法を導入する。

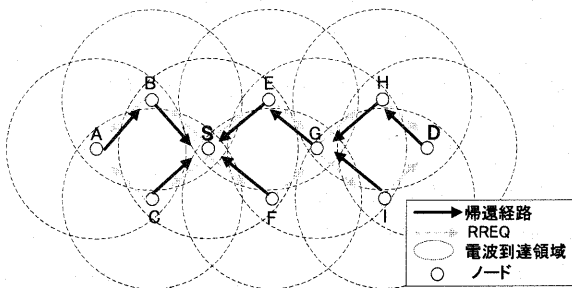
2. AODV の概要と従来研究

2.1. AODV の概要

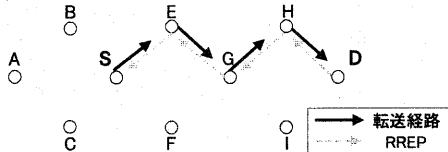
送信元 S は、宛先 D への経路が必要になったとき経路発見を開始する。経路発見は、ネットワーク全体への Route Request メッセージ (以下、RREQ と呼ぶ) の広報と Route Reply メッセージ (以下、RREP と呼ぶ) のユニキャストからなる。

(1) RREQ と帰還経路

S は D に至るための次ホップのノード (以下、転送経路と呼ぶ) の発見を要求するためにブロードキャストする RREQ は、それを受信したノードに対して S に至るための次ホップのノード (以下、帰還経路と呼ぶ) を与えることになる。例えば、図 1(a) の E は S から RREQ を受信することで帰還経路を S とし、G は E から RREQ



(a) RREQ による帰還経路の作成



(b) RREP による転送経路の作成

図1 RREQとRREPによる経路の作成

```

if (receiving RREQ and broadcast_idd ≠ broadcast_idd) or
receiving RREP then (1)
(2)
if sequnumd < sequnumj or (3)
(sequnumd = sequnumj and hopcountd > hopcountj) then (4)
sequnumd = sequnumj; (5)
hopcountd = hopcountj + 1; (6)
nexthopd = j; (7)
endif (8)
endif (9)

```

図2 AODVの経路更新規則

を受信することで帰還経路をEとする。RREQを受信したノードがDへの経路を取得していない場合、受信したRREQのブロードキャストを行う。なお、各ノードはシーケンス番号及びブロードキャスト番号と呼ばれる自身の番号をそれぞれ管理する。シーケンス番号は、経路の循環を防止し新しい経路を与えるために用いられ経路発見を行うたびにSは自身のシーケンス番号に1増加した値をRREQに指定する。SがブロードキャストしたRREQを受信する各ノードは、先に受信したRREQに示されるSのシーケンス番号に比べ後に受信したRREQに示されるSのシーケンス番号が大きい場合に帰還経路を更新する。また、ブロードキャスト番号は、各ノードが先に受信したRREQと後に受信したRREQが重複するかどうかを判断するために用いられ経路発見を行うたびにSは自身のブロードキャスト番号に1増加した値をRREQに指定する。各ノードは、先に受信したRREQに示されるSのブロードキャスト番号と後に受信したRREQに示されるSのブロードキャスト番号が同一の場合、重複するRREQと判断し後に受信したRREQによる帰還経路の作成及び再ブロードキャストを行わない。例えば図1(a)GのようにEからRREQを受信した後にFからRREQを受信した場合、RREQに示されるSのブロードキャスト番号が同一であることから重複するRREQと判断し直ちに破棄する。

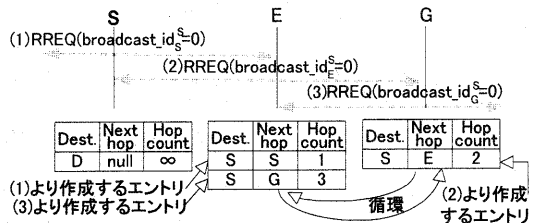


図3 重複するRREQにより生じる循環の例

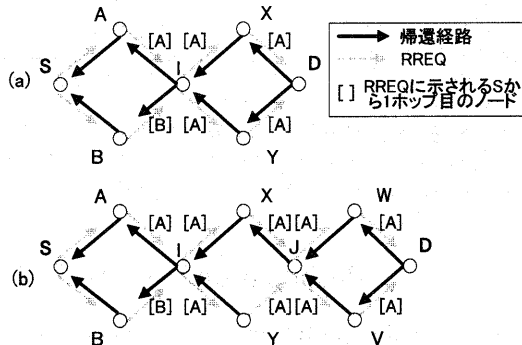


図4 従来研究[3]における経路の作成例

(2)RREPと転送経路

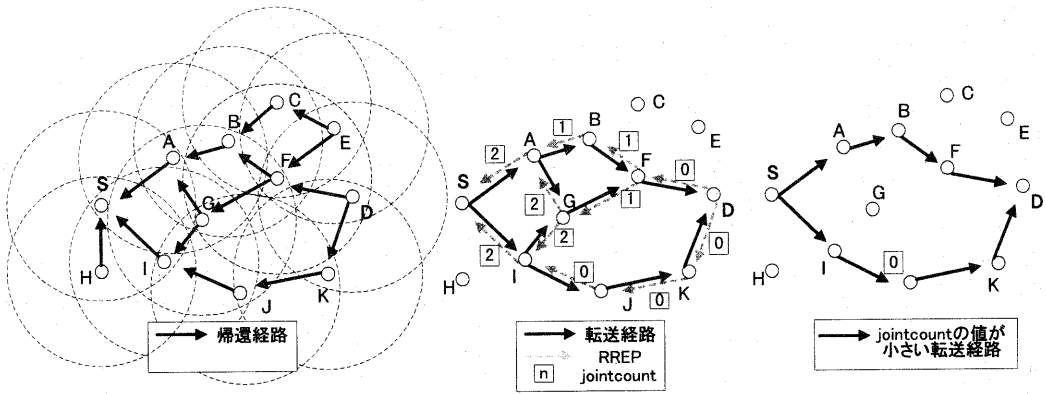
D自身もしくは既に転送経路を取得しているノードがRREQを受信した場合、RREPをSに向かってユニキャストする。RREPは作成済みの帰還経路に沿ってSに至る。RREPは、それを受信したノードに対して転送経路を与えることになる。例えば図1(b)のHはDからRREPを受信することで転送経路をDとし、作成済みの帰還経路GにRREPを送信する。DからSに至るための帰還経路は1つとなり(図1(a))、その帰還経路に沿ってRREPがSに至る結果SからDに至る転送経路は1つとなる(図1(b))。

図2にノードiがjからRREQもしくはRREPを受信するたびに行うAODVの経路更新規則を示す。

$broadcast_id^d$, $sequnum^d$, $hopcount^d$, $nexthop^d$ はそれぞれiにおける宛先dに関するブロードキャスト番号、シーケンス番号、ホップ数、次ホップを表す。iは先に受信したRREQとブロードキャスト番号が同一のRREQを受信した場合は、経路の更新を行わない(図2(1))。受信したRREQもしくはRREPのシーケンス番号がより大きい場合に経路を更新し(図2(3))、シーケンス番号が同一であってもホップ数がより小さい経路となる場合に経路を更新することで最短経路を求める(図2(4))。iは先に取得したdのシーケンス番号をjから受信したメッセージに示される値に更新し(図2(5))、dに至るためのホップ数をjから受信したメッセージに示されるホップ数に1を加えた値に更新する(図2(6))。そして、dに至るための次ホップをjに更新する(図2(7))。

(3) エントリの有効期限の更新方法

(1)(2)により得た帰還経路と転送経路は、それぞれルーチング・テーブルのエントリとして各ノードで保持する。エントリには有効期限が与えられ、データパケットを転送するたびに、その転送に用いたエントリの



(a) 複数の帰還経路の作成 (b) 複数の転送経路の作成 (c) リンクが重複しない経路の選択

図5 リンクが重複しない経路を求める計算手順

有効期限を延ばす。例えばRREPが通過しなかった図1(a)のAの帰還経路Bのようにデータパケットの転送に用いられないエントリは、有効期限後に無効となる。

(4)経路の切断とRERR

各ノードは、隣接ノードに定期的にHelloメッセージを通知する。データパケットの転送を行うノードが次ホップの隣接ノードからHelloメッセージを受信しなくなった場合やMAC(Medium access control)層から送信失敗通知があった場合、RERR(Route Error)メッセージをSに向かって送信する。

2.2. マルチパス・ルーチングの従来研究

オンデマンド型のルーチング・プロトコルが経路を作成するために用いる経路発見を行う回数を抑制することで、ルーチングに用いるメッセージ数の減少と経路発見に要する時間を短縮し、その結果、伝送可能なデータパケット数の増加と伝送遅延の短縮が可能となる。そこで、AODVを拡張し1回の経路発見で複数の経路を作成するマルチパス・ルーチング・プロトコルがこれまでに提案されている[2][3]。

[2]は、Sのブロードキャスト番号が重複するRREQについても帰還経路を作成することで複数の経路を作成する。例えば図1(a)のGは作成済みの帰還経路Eに加えFから受信したSのブロードキャスト番号が重複するRREQにより帰還経路Fを作成し、作成した複数の帰還経路上をRREPが転送されることで複数の転送経路を得る。しかしながら、ブロードキャスト番号が重複するRREQによる帰還経路の作成は隣接ノード間で循環する経路を作成してしまう。これは循環する経路でデータパケットが循環し宛先に至らないのみならず輻輳の原因となってしまう問題がある。図3のEはSからのRREQ(図3(1))を受信した後にGから受信したSのブロードキャスト番号が重複するRREQ(図3(3))により帰還経路Gを作成する。図3のE及びGのルーチング・テーブルのエントリからEとG間で循環が生ることがわかる。

[3]は、RREQに追加拡張したSから1ホップ目のノードを示すフィールドを用いることでSとD間に互いにリンクが重複しない経路を作成する方法を提案している。例えば図4(a)のIは、Aから受信したRREQにより帰還経路Aを作成した後、Bから受信したRREQはそのフ

ィールドの値がBであり先にAから受信したフィールドの値Aと異なるため帰還経路Bを作成する。これより、SとI間に互いにリンクが重複しない帰還経路I-A-S及びI-B-Sを得る。AODVと同様に、IはBから受信したSのブロードキャスト番号が重複するRREQの再ブロードキャストを行わない。Dは異なる隣接ノードからRREQを受信した場合、それぞれのRREQにより帰還経路を作成することでIとD間の経路でノードが重複しない帰還経路D-X-I及びD-Y-Iを得る。SとI間及びIとD間で互いにリンクが重複しない帰還経路上にRREPが送信されることで、SとD間で互いにリンクが重複しない転送経路を得る。

しかしながら、このようにS及びDから1ホップ目のノードの情報を用いる方法では中継ノードが2つ以上共通する場合にSとD間で互いにリンクが重複しない経路を作成できなくなってしまう問題がある。例えば図4(b)のようにSとD間にリンクが重複しない経路を作成できなくなってしまう問題がある。図4(b)のIは、先にAから受信したRREQのブロードキャストを行い、後にBから受信した重複するRREQの再ブロードキャストを行わないことからX、Y及びJなどのI以降のノードが受信するRREQはフィールドの値Aを示すものとなる。これより、JがX及びYから受信するフィールドの値は同一のAとなるため後に受信したRREQによる帰還経路の作成は行わない。図4(b)はJが先にXからRREQを受信しその後Yから受信する例であり、Jは帰還経路Yを作成しない。その結果、Jは帰還経路XのみにRREPを送信することになるためIとJ間の経路は2つの経路を作成できるにも関わらず得られるのは1つの経路となり、IとJ間のリンクが重複する経路になってしまう。

3. マルチパス・ルーチングの提案

提案するルーチングはAODVを拡張することで実現し、循環のない複数の経路の作成を行い、かつ、データパケットの転送に用いる経路(以下、アクティブ経路と呼ぶ)と予備とする経路(以下、バックアップ経路と呼ぶ)が互いにリンクが重複しないことでアクティブ経路の切断に対する耐障害性を図る。

アクティブ経路は、AODVと同様にデータパケットを転送するたびに有効期限を延ばすことで維持しパッ

クアップ経路は 3.2 節で述べる方法により維持する。提案するルーチングは、以下に示す 2 つの構成からなり次節でその詳細を述べる。

- (1) 互いにリンクが重複しない経路の計算手順
- (2) 複数経路の維持

3.1. 互いにリンクが重複しない経路の計算手順

S と D 間に互いにリンクが重複しない経路を求める計算手順は、先ず複数の経路の作成を行い、次にそれらの複数経路の中から特定の規則に従って経路を選択する手順となる。

(1) 複数の経路の作成

複数の経路を得るためには 1 つの宛先に対して複数の帰還経路を作成することが必要となる。そこで、AODV のように最初に受信した RREQ により帰還経路を作成し、その後受信した S のブロードキャスト番号が重複する RREQ を直ちに破棄するのではなく、重複する RREQ についても帰還経路の作成を行う。ただし、隣接ノード間での循環を防止するため受信した RREQ に示される S までのホップ数がルーチング・テーブルの S のエントリに示されるホップ数よりも小さい場合に帰還経路に加える。例えば図 5(a) の G は、A から受信した RREQ (図 7(1)) により帰還経路 A を作成する (図 7(2))。その後 I から受信した S のブロードキャスト番号が重複する RREQ (図 7(3)) に示される S までのホップ数 1 が、エントリに示される S までのホップ数 2 (図 7 斜線部) よりも小さいことから I から受信した RREQ による帰還経路 I を作成する (図 7(4))。G は F から受信した重複する RREQ に示されるホップ数はエントリのホップ数よりも大きいため帰還経路の作成は行わず破棄することで図 3 のような循環を防止する (図 7(6))。

(2) RREP の制御

例えば図 5(b) において B から D に至る次ホップは F であり、G から D に至る経路は F である。F は D に至るための経路が交差するノード (以下、交差ノードと呼ぶ) となる。以下の (3) においてリンクが重複しない経路を求めるために交差ノード数を用いることから、AODV の RREP に交差ノード数を表すフィールド (以下、jointcount と呼ぶ) を追加拡張する。交差ノードは、例えば図 5(b) の G のように作成済みの複数の帰還経路に RREP の送信を行うノードとなることから RREP を受信したノードが複数の帰還経路を持つ場合に jointcount の値を 1 増加させる。なお、D は常に jointcount の値を 0 として RREP をユニキャストする。

例えば図 5(b) の F は複数の帰還経路を持つことから、D から受信した RREP の jointcount の値に 1 を加えた RREP を B 及び G に送信する。同様に複数の帰還経路を持つ G は、F から受信した RREP の jointcount の値に 1 を加えた RREP を A 及び I に送信する。なお、例えば図 5(b) の A や I は複数の RREP を受信することになる。このとき、先に受信した RREP の jointcount の値よりも後に受信した RREP の値が大きい場合は、後に受信した RREP の送信を行うことで帰還経路上にあるノードにあらたに取得した交差ノード数を提供する。先に受信した RREP の jointcount の値よりも後に受信した RREP の値が小さい場合、及び同一の値の場合

```

if sequnumid < sequnumjd then (1)
  entryid = {d | destination = d and (2)
    (nexthopid, hopcountid, sequnumid) ∈ entryid};
  entryid = NULL; (3)
create entryid = (j, hopcountjd + 1, sequnumjd); (4)
else (5)
  entryid = {d | destination = d and (6)
    (nexthopid, hopcountid, sequnumid) ∈ entryid};
  max_hopcountid = maxk{hopcountkd | (7)
    (nexthopkd, hopcountkd, sequnumkd) ∈ entryid};
  if sequnumid = sequnumjd and max_hopcountid > hopcountjd then (8)
    create entryid = (j, hopcountjd + 1, sequnumjd); (9)
  endif (10)
endif (11)
  
```

図 6 エントリの作成規則

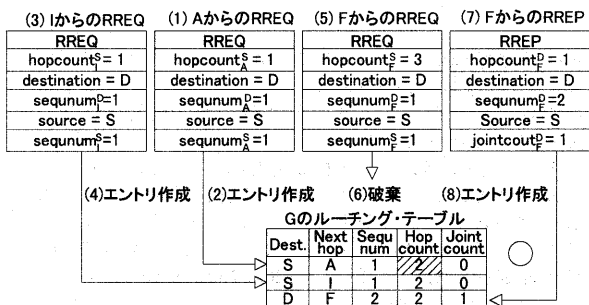


図 7 G におけるエントリの作成例

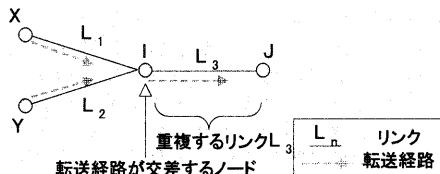


図 8 重複するリンクとなる例

```

entrykd = {jointcountkd | min(jointcountkd) and (1)
  (nexthopkd, hopcountkd, sequnumkd, jointcountkd, lifetimekd) ∈ entryid}; (2)
if |entrykd| = 1 then (3)
  select_entry := entrykd; (4)
else (5)
  select_entry := {lifetimeid | max(lifetimeid) and (6)
    (nexthopkd, hopcountkd, sequnumkd, jointcountkd, lifetimekd) ∈ entrykd}; (5)
endif (6)
  
```

図 9 経路の選択規則

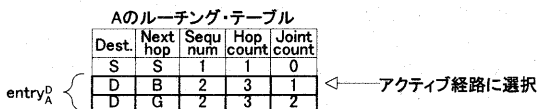


図 10 A におけるアクティブ経路の選択例

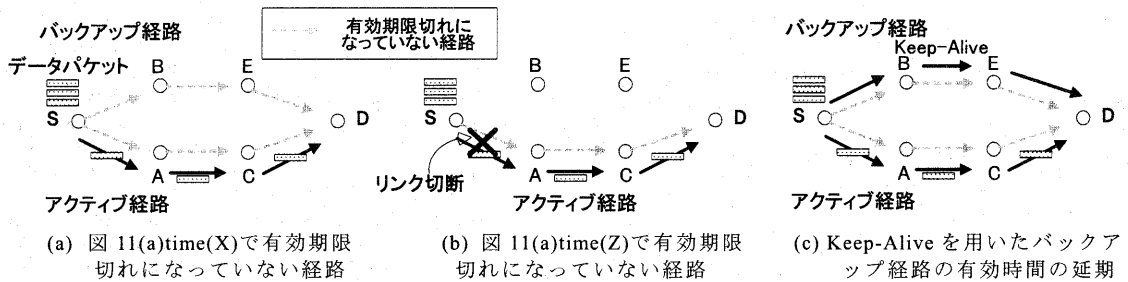


図 11 AODV の有効期限の更新方法(a)(b)と Keep-Alive を用いた有効期限の維持方法(c)

合は、冗長な RREP の送信を抑制するため後に受信した RREP の転送は行わない。

図 6 は、 i が j から RREQ もしくは RREP を受信するたびに行うエントリの作成規則である。 $entry_i^d$ は、 i における宛先 d のためのエントリの集合を表す。新しい経路を得る 2.1(1)で述べた AODV の方針を引き継ぎ、受信した RREQ もしくは RREP のシーケンス番号がエントリに示されるシーケンス番号より大きい場合、全てのエントリを無効とし受信したメッセージによりエントリを作成する (図 6(1-4))。また、図 5(a)の G のようにシーケンス番号が同一のメッセージを受信し、エントリに示されるホップ数よりもメッセージに示されるホップ数が小さい場合のみ受信したメッセージによりエントリを作成する (図 6(5-9))。

(3) リンクが重複しない経路の選択

互いにリンクが重複しない経路は、(2)により得た複数の転送経路の中から特定の規則に従って経路を選択することで計算する。一般にリンクが重複する経路は、図 8 に示すように 2 つの転送経路が交差するときに生じ、交差ノード以降のリンクが重複するリンクとなる。従って複数の転送経路を持つノードが、他のノードの転送経路と交差することが少ない転送経路を選択することで、経路が交差しリンクが重複してしまうことを回避できる。なお他のノードの転送経路と交差する回数は、各ノードから D に至るまでに存在する交差ノードの数となる。

例えば転送経路を示した図 5(b)において交差ノードは、 G 、 F 及び D となる。 A には 2 つの転送経路 $B-F-D$ 及び $G-F-D$ があり、それぞれの経路上にある交差ノードはそれぞれ (F, D)、(G, F, D) である。従って、 A は交差ノード数がより小さい転送経路 $B-F-D$ を選択する。同様にして I は転送経路 $J-K-D$ を選択する。各ノードがこのようにして選択した転送経路は、 $S-A-B-F-D$ 及び $S-I-J-K-D$ となり、図 5(c)より互いにリンクが重複しない経路となることわかる。

図 9 に交差ノード数を用いることで S と D 間に互いにリンクが重複しない経路を求めするための経路の選択規則を示す。 $jointcount_i^d$ と $lifetime_i^d$ は、それぞれ交差ノード数及び有効期限を表す。図 9(1-3)は、ノード i において宛先 d のためのエントリの中から交差ノード数が最も小さいエントリを選択することを示している。交差ノード数が最も小さいエントリが複数ある場合は、最も有効期限が長い経路を選択する (図 9(5))。この選択規則は、転送経路が作成されるたび、及び選

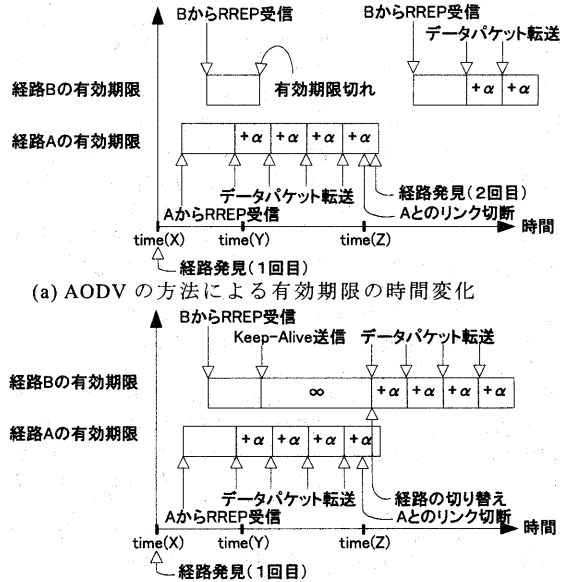


図 12 S (図 11) における経路 A 及び B の有効期限の時間変化の比較

択した転送経路が切断されるたびに行う。図 10 に A における経路の選択例を示す。

3.2. 複数経路の維持

3.1 節で求めたリンクが互いに重複しない経路は、そのエントリを作成後有効期限が時々刻々と短くなっていく。2.1(3)で述べた AODV の有効期限の更新方法を用いる場合、例えば図 11(a)のように S がアクティブ経路に A を選択してデータパケットの転送を開始してから図 12(a)に示すように S のエントリ有効期限が変化する。図 12(a)のように $time(Z)$ で S と A 間のリンクが切断した場合、 $S-B-E-D$ のリンクは接続しておりデータパケットを転送できるにも関わらずバックアップ経路 B は有効期限切れになっていることから S は経路発見を再び行ってしまふ (図 11(b))。

そこで例えば、アクティブ経路及びバックアップ経路の両方を長い有効期限に設定する方法がある。しかしながら有効期限を長くした場合、 S のバックアップ経路の有効期限が長く維持されることから S は既に転送先で切断されているバックアップ経路に対してもデ

文 献

- [1] C.E.Perkins and E.M.Royer, "Ad Hoc On-demand Distance Vector Routing", In Proc. of 2nd IEEE Wksp. Mobile Comp. Sys. and Apps., 1999.
- [2] M. Jiang and R. Jan, "An Efficient Multiple Paths Routing Protocol for Ad-Hoc Networks", In Proc. of the International Conference on Information Networking(ICOIN), 2001.
- [3] M.K.Marina and S.R.Das, "On-demand Multipath Distance Vector Routing for Ad Hoc Networks", In Proc. of the International Conference for Network Protocols(ICNP), 2001.
- [4] A. Nasipuri, R. Castaneda, and S. R. Das, "Performance of Multipath Routing for On-demand Protocols in Mobile Ad Hoc Networks", In Proc. of ACM/Kluwer Mobile Networks and Applications(MONET), pp.339-349, 2001.
- [5] Y. Hu and D.B.Johnson, "Caching Strategies in On-Demand Routing Protocols for Wireless Ad Hoc Networks", In Proc. of the International Conference on Mobile Computing and Networking(Mobicom), 2000.

ータパケットを転送してしまう問題があることが経路の有効期限に関する研究により報告されている[5].

これより, AODVの有効期限の更新方法や単に有効期限を長くする方法ではバックアップ経路を有効利用することが困難となる. そこで, バックアップ経路の有効期限を維持するためのあらたな方法が必要となり, バックアップ経路の有効期限を維持するためのメッセージ(以下, Keep-Aliveと呼ぶ)を導入する.

Sは最初に受信したRREPにより作成した転送経路を用いて直ちにデータパケットの転送を開始する. その後RREPを受信しあらたな転送経路が作成された場合, 図6の経路の選択規則に従い先ずデータパケットの転送に用いるアクティブ経路を決定しデータパケットの転送を開始する. 次にアクティブ経路に選択されなかった残りの他方の経路に対してSはKeep-Aliveをユニキャストする(図10(c)).

例えば図11(c)のKeep-Aliveを受信したBは, 転送経路Eを用いるデータパケットを受信しない場合であっても, Eとリンクが接続している限り転送経路Eの有効期限を延ばす. Bは, 受信したKeep-AliveをEに転送する. Keep-Aliveを受信した経路はバックアップ経路として認識される. なお, 転送経路が複数ある場合は, 既にデータパケットの転送に用いていない残りの転送経路の中から図6の規則に従い1つの経路を選ぶ. バックアップ経路が利用不可となり, かつ, 有効な転送経路が残っていない場合, RERRをSに向かって送信する.

図12(b)に示すようにKeep-Aliveを用いることでバックアップ経路Bの有効期限がSにおいて保たれる. time(Z)でSとA間のリンクが切断した場合, 図11(b)に示すようにSにおいてバックアップ経路Bの有効期限が保たれているためSはバックアップ経路Bにデータパケットの転送先を切り替えることが可能となり図11(a)のように経路発見を再び行う必要はない.

4. おわりに

本稿では, アドホックネットワークのためのマルチパス・ルーチングを提案した.

経路が切断するたびに行う頻繁な経路発見を抑制することでオンデマンド型のルーチング・プロトコルの性能を向上するためこれまでにAODVを拡張したマルチパス・ルーチング・プロトコルが提案されていた. しかしながら, 作成した経路が隣接ノード間で循環してしまう問題や経路の切断に対して耐障害性を提供する重複しない経路の作成を行うことができなくなってしまう問題があった.

提案するルーチングは, データパケットの宛先に至るための経路が交差するノードに着目することで互いにリンクが重複しない経路の作成する. 作成した一方の経路をデータパケットの転送に用いる経路とし, 残りの他方の経路には予備とすることを通知するメッセージを送信することでその有効期限を保つための対策を行った.

今後は, 提案したマルチパス・ルーチングと従来研究をシミュレーションにより比較評価することが課題である.

謝辞

日頃ご指導頂く(株)KDDI研究所浅見所長, 松島副所長, 水池取締役に感謝する.