03-01048

アドホックネットワークにおける階層ルーティングの実装と応用（継続）

代表研究者　角田良明　広島市立大学情報科学部教授
共同研究者　井上伸二　広島市立大学情報科学部助手
〃　大田知行　広島市立大学情報科学部助手

1　アドホックネットワークテストベッドフレームワークの開発

1.1　はじめに

アドホックネットワーク[1]とは，基地局を必要としないモバイルノード（以降，ノードと記す）の集まりによって形成されるネットワークである。各ノードはルータの機能を持ち，直接通信できない離れたノードとは途中のノードを中継する無線マルチホップ通信[2]により通信が可能となる。アドホックネットワークはその特性により，その利用分野は多岐に広がりを持つ。例えば，限られた範囲での簡易なネットワークの構築や，災害時の即席なネットワークの構築などが挙げられる。よって，アドホックネットワークはこれからのネットワークに重要な存在となると考えられる。

モバイルネットワークの使用例として携帯電話機能がある。遠く離れた人同士が快適に会話を行うためには，接続性の高さが必然的に要求される。アドホックネットワークでは，ノードの移動によるネットワークトポロジの頻繁な変化などの厳しい条件がある。つまりアドホックネットワークではこのような条件に適応しながら，高い接続性を保つことが重要な課題となる。このことから，アドホックネットワークにおいて，通信中の高い接続性を保つために様々なルーティングプロトコルが提案されている。

しかしながら，提案されているルーティングプロトコルのほとんどがシミュレーションによる検証で有効性が確認されているだけである。実環境には，シミュレーションでは想定していないような影響や，問題が存在する。例えば，他の電子機器との電波干渉や，開けた場所での電波の拡散などが挙げられる。今後，アドホックネットワークの実用化に向けての開発を考えると，シミュレーションだけではなく，実機における実験や検証が必要であると考えられる。しかしながら，そのためには実機におけるアドホックネットワークの実験環境が必要である。

そこで，アドホックネットワークテストベッドフレームワークを開発した。これを AdHocEngine と呼ぶ。本研究では，実機におけるアドホックネットワークテストベッドフレームワークの開発として，次のことを行った。一つ目は，ルーティングフレームワークの開発である。これにより，容易にルーティングプロトコルを実装，交換できるようになる。二つ目は，エンドツーエンド通信の信頼性向上のための確認応答機能の開発である。

本研究で開発したソフトウェアは，Microsoft VisualC＋＋.NET を使用して開発を行い，Windows XP 上で動作する。また，無線通信デバイスとして無線 LAN（IEEE802.11b）を用いる。

本稿は，2 節でアドホックネットワークテストベッドフレームワークについて述べる。3 節では開発した AdHocRouting フレームワークについて述べ，4 節では開発した確認応答機能について述べる。また，5 節では AdHocRouting フレームワークの動作確認のための通信実験とその結果に対する考察について述べ，6 節では確認応答機能の性能評価のための通信実験の説明と結果に対する考察を述べる。最後に 7 節で本研究のまとめと今後の課題について述べる。

1.2　アドホックネットワークテストベッドフレームワーク

AdHocEngine は，アプリケーションにアドホックネットワーク上でのマルチホップ通信を提供している。また，AdHocEngine は，DLL（Dynamic Link Library）として実装している。この AdHoc Engine を使用することで，他のノードとの通信，及びノード間通信の際のパケットの中継が可能となる。

AdHocEngine は**図 1** に示すようなフレームワークで実装している。AdHocEngine の主なレイヤの説明を以

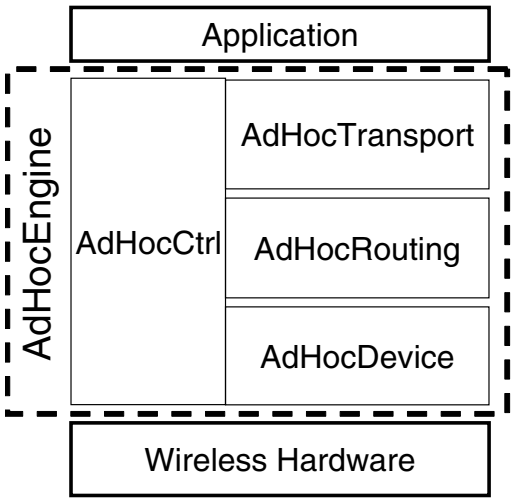

図 1 ：AdHocEngine フレームワーク

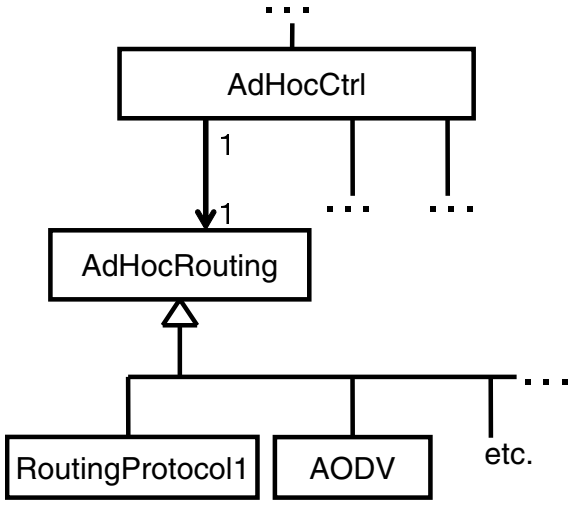

図 2 ：クラス図

下に記す。

AdHocCtrl　データグラム型の通信機能，パケットの中継機能などを，複数の AdHocTransport，AdHocDevice を管理しつつ提供する。パケットの送信では，AdHocRouting からの情報を元に次ホップノードを選択し，送信する。

AdHocTransport　アプリケーションに対して，データ通信のインタフェースを提供する。AdHocCtrl に対して複数のプロトコルを設定することが可能である。

AdHocRouting　AdHocCtrl にルーティング情報を提供する。抽象クラスとして定義されているため，このクラスを継承することにより，様々なルーティングアルゴリズムが実装できる。

AdHocDevice　無線 LAN（IEEE802.11b）インタフェースを提供し，隣接するノード同士のデータの送受信を実装しているクラス。

本研究では，AdHocRouting を主に開発し，ルーティングフレームワークを実装した。また，AdHoc Transport において確認応答機能を実装している。

1.3　AdHocRouting フレームワーク

AdHocRouting 周辺のクラス図を**図 2** に示す。全ての送信，受信パケットはその種類（データパケットや制御パケット）に関わらず必ず AdHocRouting に渡される。これにより，どのようなルーティングプロトコルを実

装した場合でも対応しやすくなっている。また，AdHocRouting には AdHocCtrl への受信処理要求や，重複したパケットを処理するためのパケットキャッシュなど，どんなルーティングプロトコルにも共通に必要な機能を実装している。各ルーティングプロトコルはこの AdHocRouting を継承し，実装される。複数のルーティングプロトコルが実装された場合，AdHocCtrl でどれを用いるか指定している。この AdHocCtrl 中の指定を変更することでルーティングプロトコルを容易に変更できる。

以下に実装した AdHocRouting フレームワークの動作の簡単な流れを記す。

STEP 1　AdHocCtrl に届いた受信パケット，または AdHocCtrl で作成された送信パケットは AdHocRouting へ渡される。

STEP 2　AdHocRouting は受け取ったパケットが過去に受け取ったことのある重複したパケットでないかをパケットキャッシュを見て判断し，重複したものの場合破棄する。

STEP 3　初めて受け取ったパケットであれば，パケットの送信先を確認し，自分宛てのデータパケットであれば AdHocCtrl に受信処理要求を出す。

STEP 4　パケットが他のノード宛てであった場合，指定されたルーティングプロトコルによって次ホップノードのアドレスを求め，AdHocCtrl へ次ホップノードのアドレスとパケットを渡し，送信処理要求を出す。

本研究では，AODV[3]をルーティングプロトコルとして実装し，動作確認を行った。

1.4　AdHocTransport フレームワーク

1.4.1　エンドツーエンド確認応答機能

マルチホップ通信では，他のノードを中継して通信を行うため,ホップ数が増加する。無線通信は，天候や，周辺の機器などの外部からの雑音を受けやすい。そのため，ホップ数の増加に伴いパケットが損失する可能性が高くなる。また，メディアアクセス制御方式[4]を用いるだけでは信頼性を確保することが困難となる。さらに，ノード数が増加すると，複数のノードが同時にパケットを送信してしまい，パケットが衝突し損失してしまう。このため，エンドツーエンド通信での信頼性が低下し，確実な通信ができなくなる。そこで，エンドツーエンド通信での信頼性を向上させるために，確認応答機能の開発を行った。

ノードAからノードCへの通信における，エンドツーエンド確認応答機能を用いた通信例を**図 3** に示す。以下にその説明を示す。

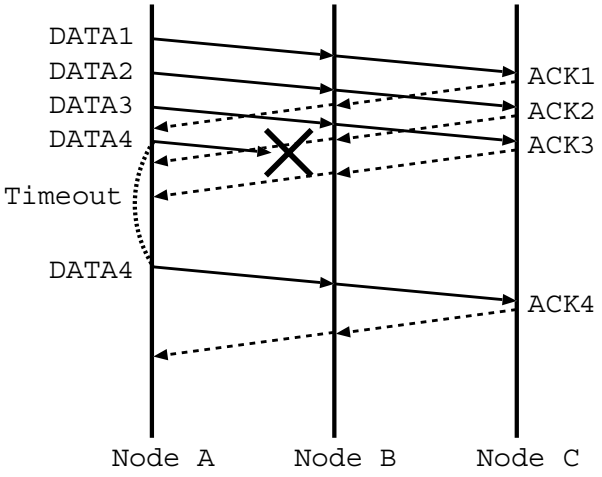

図 3 ：エンドツーエンド確認応答機能

1．ノードAは DATA1 を送信する。この時，同時に時間 T のタイマをスタートさせ，与えられた送信間隔で DATA2，DATA3，DATA4 を送信する。

2．ノードBは DATA1 を受信し，宛先アドレスが自分宛でないので，ノードCに DATA1 を転送する。

3．ノードCは DATA1 を受信すると，宛先アドレスが自分宛なので，上位層にデータパケットを渡したあ

と，ACK1 をノードAに返信する。

4．DATA4 で示されるように，タイムアウト内にノードCからの ACK4 が返ってこない場合，DATA4 は途中で失われたと判断して，ノードAは DATA4 を再送する。

5．パケットの重複はシーケンス番号によって判断され，ノードCはすでに受信しているパケットを受信した場合，ACK が失われたと判断し，そのパケットを捨て，ACK を再びノードAに返信する。

エンドツーエンド確認応答機能では，ACK を待たずパケットの送信を行うので，受信側でのパケットの順序誤りが起ってしまう。このため，受信パケットの順序を入れ替える処理が必要であり，今後実装する予定である。

1.4.2 1ホップ確認応答機能

RTS/CTS 方式を使用すると，RTS フレームや CTS フレームの送信を行うため，非常に帯域を消費してしまう。無線アドホックネットワークでは，ネットワークトポロジの変化に伴うルーティングや，隣接ノードを探索する場合のような，制御情報を交換する際には，ネットワーク中に多くのパケットが送信されるため，帯域をできるだけ節約しなくてはならない。そのため，ブロードキャストアドレスを利用して通信を行うことがある。しかしながら，ブロードキャストを利用した通信ではノード間での通信における信頼性を確保できない。そこで，本研究では，ルーティングでの制御パケットの各ノード間での通信における信頼性を向上させるため，1ホップ確認応答機能を提案する。

ノードAからノードCへの通信における，1ホップ確認応答機能を用いた通信例を**図 4** に示す。以下にその説明を示す。

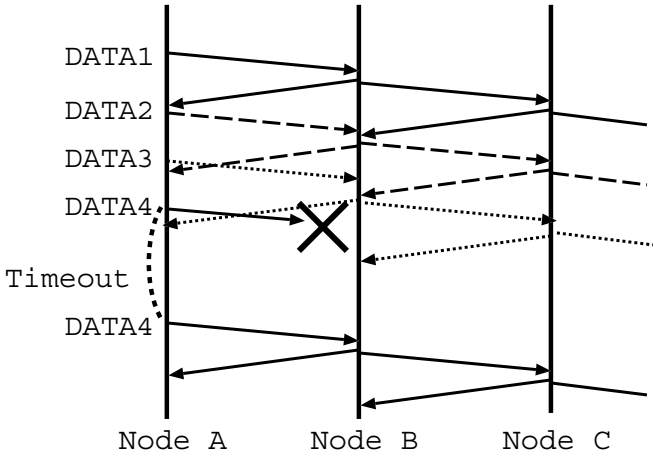

図 4 ： 1 ホップ確認応答機能

1．ノードAは DATA1 を隣接ノードに対してブロードキャストする。この時，同時に時間 T のタイマをスタートさせ，与えられた送信間隔で DATA2，DATA3，DATA4 を送信する。

2．ノードBは DATA1 を受信すると，DATA1 の宛先アドレスが自分宛でないので，DATA1 をブロードキャストで隣接ノードへ転送する。

3．ノードAは DATA1 をノードBから傍受すると，それをノードBからの ACK と判断して，DATA1 がノードBへ到達できたことを確認できる。もし，DATA4 で示されるようにタイムアウト内に ACK を受信しない場合，ノードAは DATA4 を再送する。

4．ノードCは DATA1 の宛先アドレスが自分宛だが，隣接ノードに DATA1 をブロードキャストする。これにより，ノードBもノードCへのパケットの到達を確認することができる。

5．パケットの重複はシーケンス番号によって判断され，ノードCはすでに受信しているパケットを受信した場合，ACK が失われたと判断し，そのパケットを捨てて，ACK を再びノードBに返信する。

アドホックネットワークでは，ノードがパケットを送信する際，外部からの雑音により，パケットを中継している途中でパケットが損失することがある。よって，送信する回数が多くなるに従い，パケットを損失する可能性が高くなる。そこで，パケットを再送しない状況において，送信元ノードから宛先ノードへのホップ数が L

である経路での，各機能の必要なパケット送信回数を考える。エンドツーエンド確認応答機能の場合，パケットを送信して ACK が返ってくるまでの送信回数は，1パケット当たり $2(L-1)$ であるのに対し，1ホップ確認応答機能の場合は，宛先ノードまでの L 回となる。L が大きくなるほど，エンドツーエンド確認応答機能に比べて，1ホップ確認応答機能では，パケットの送信回数を減らすことができる。また，パケットを中継している途中でパケットが損失した場合，1ホップ確認応答機能では，送信元ノードよりも宛先ノードに近いノードから再送が行えるため，再送するパケットのホップ数を削減できる。したがって，1ホップ確認応答機能は，送信元ノードから遠い宛先ノードに対してのパケットを送信する場合に有効であると考えられる。

1.5 通信実験1

1.5.1 実験目的

実装した AdHocRouting フレームワークの動作確認として，Flooding と AODV のルーティングプロトコルの性能評価を実機において行う。Flooding との性能比較を行う場合，送信先への経路が1つであるネットワークトポロジでは経路探索や経路維持の必要性がないため優劣がつきにくいと考えられる。従って，複数の経路が存在するようにノードを配置し，通信実験を行った。

1.5.2 実験方法

ルーティングプロトコルが Flooding と AODV の場合それぞれに対して，**図5**のようにノードを配置し，ノードSからノードDにデータパケット（データペイロードは1024バイト）を1個，10個，50個，100個送信する。複数のパケットを送信する場合は，送信間隔なしに次々と送信する。ノードDは受信したデータパケットそれぞれに対して ACK パケットをノードSに返信する。この時の送信データパケット数に対するノードDが受信したデータパケット数の割合（以降，データ到達率と記す）と，送信データパケット数に対するノードSが受信したACKパケット数の割合（以降，ACK 到達率と記す），ノードSにおいて最初のデータパケットを送信した時から最後の ACK パケットが到達した時までの時間（以降，応答時間と記す）を計測する。それぞれの場合において10回ずつの平均を取った。

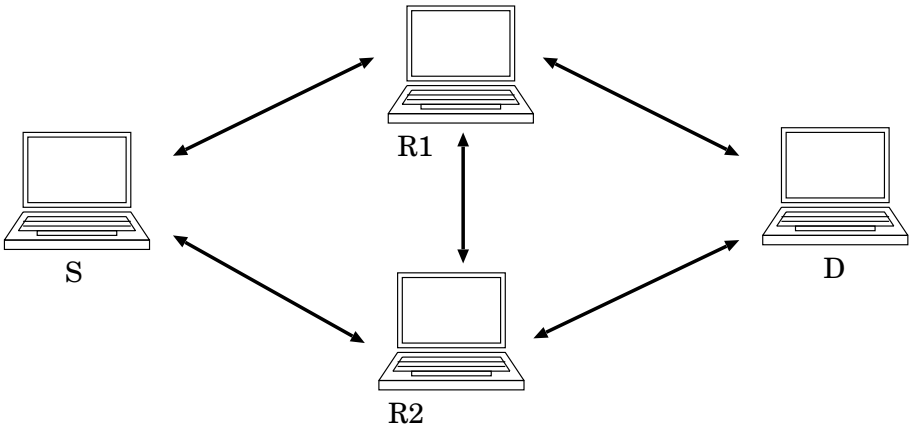

図5：通信実験におけるネットワークトポロジ

1.5.3 実験結果および考察

図6にデータ到達率，**図7**に ACK 到達率，**図8**に応答時間の測定結果をそれぞれ示す。また，図の横軸はノードSが送信したデータパケット数を示す。

図6の Flooding のデータ到達率に注目してみると，送信パケット全体の約1割程のパケット損失が生じた。また，**図7**の ACK 到達率に注目すると，データ到達率からさらに約1割程のパケット損失が生じている。これは，Flooding では R1 と R2 が共に中継をするため，複数のノードが同時にパケットを送信する可能性が高くなる。そのため，電波の衝突が生じてしまい，パケットの消失が多く発生しているからと考えられる。一方，

AODV のデータ到達率と ACK 到達率はほぼ100%である。AODV では Flooding と異なり通信経路を生成しているため，中継するノードは R1 か R2 のどちらか一方になる。そのため，複数のノードが同時にパケットを送信する可能性は低くなる。従って，Flooding に比べ，電波の衝突が生じにくいと考えられる。

また，図 8 の応答時間に注目してみると，AODV が Flooding に比べ 2 倍以上速い。これは，Flooding では中継ノードがパケットを受け取ると，周辺ノードへそのパケットをブロードキャストする。本実験では中継ノードが 2 つ存在し，互いが同一のパケットを送受信しあったり，送信先ノードへ同一のパケットを送信するため，

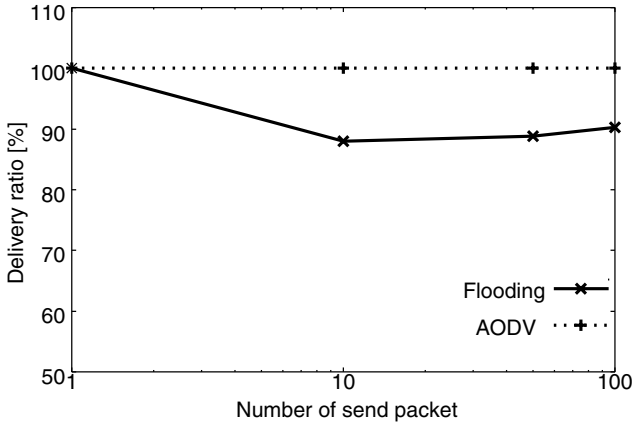

図 6 ：データ到達率の計測結果

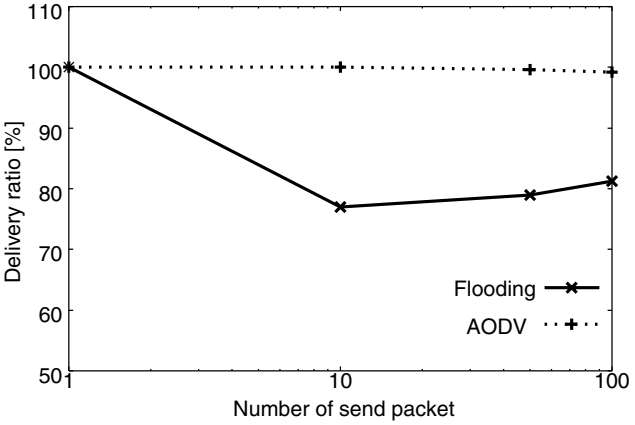

図 7 ：ACK 到達率の計測結果

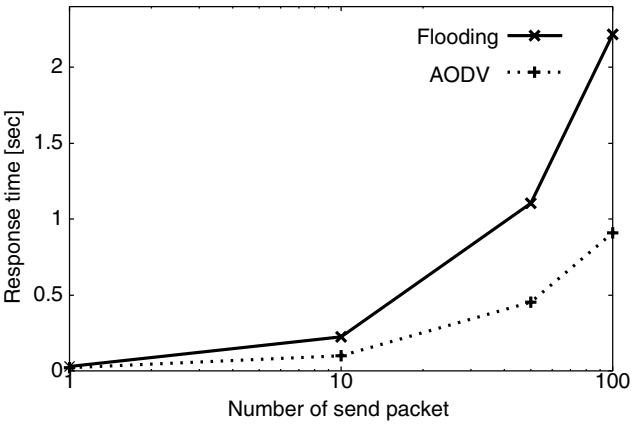

図 8 ：応答時間の計測結果

重複したパケットを処理することが度々生じる。一方，AODV ではこのような状況が発生せず，パケットの送受信の処理に無駄が生じない。従って，この処理時間の差が応答時間の差となっていると考えられる。

この通信実験の結果，ルーティングプロトコルの違いによって性能差が現れたことから，AdHocRouting フレームワークが正しく動作していることを確認した。

1.6 通信実験 2

1.6.1 実験目的

アドホックネットワークでは，直接通信が不可能なノードに対して，途中のノードがパケットを中継することにより通信が可能になる。しかしながら，複数のノードを中継することで，パケットを損失する可能性が高くなる。また，ノード数が増加するに従い，複数のノードが同時にパケットを送信すると，衝突が生じてパケットが損失してしまう可能性が高くなる。よって，エンドツーエンド間通信での信頼性が低下してしまう。そこで，本実験では，開発した確認応答機能を使用した通信実験を行い，開発した機能の性能評価を行う。

1.6.2 実験方法

エンドツーエンド通信でのパケットの到達率を計測するために，**図 9** で示すようにノードを 4 台配置する。各ノードは互いに通信範囲内に存在し，隣接ノードと直接通信することができる。また，あるノードがパケットを送信している間に，同時に他のノード同士での通信を行い，パケットがよく衝突すると思われる状態の下で通信実験を行う。

ノードAからノードDへの通信を行う際，ノードAは，途中のノードBとノードCを中継させて，ノードDへパケットを送信する。このときの，ノードDでのパケットの到達率を計測する。但し，ノードBからノードCへの通信も行い，ネットワーク中のパケットの通信量を多くする。パケットサイズによる比較を行うため，100Byte のパケットと 1000Byte のパケットの2通りの実験を行う。送信パケット数は500個，送信間隔は 50msec とする。エンドツーエンド確認応答機能では，送信元ノードが宛先ノードからの ACK の返信がないパケットに対する再送を行うまでのタイムアウトを 200msec とし，1 ホップ確認応答機能では，各ノードが隣接ノードからの ACK を受け取れないパケットに対する再送を行うまでのタイムアウトを 30msec とする。ルーティングアルゴリムは静的ルーティングを使用する。また，再送回数は 2 回までとする。

本実験には，前節で開発した確認応答機能を用い，それぞれの機能を確認機能を使用しない場合と比較し，評価を行う。

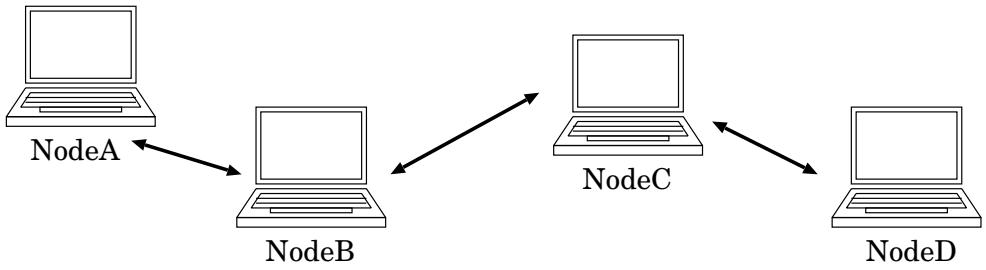

図 9 ：通信実験 2 におけるネットワークトポロジ

1.6.3 実験結果と考察

本実験の計測結果を**表 1** に示す。確認応答機能を使用せず RTS/CTS を使用した場合と，エンドツーエンド（表中は EtoE と略す）確認応答機能で RTS/CTS を使用した場合を比較すると，エンドツーエンド確認応答機能の方が高い到達率を示している。また，この結果より，RTS/CTS を使用してもパケットの損失が生じていることが考えられる。**図 9** に示すネットワークトポロジでは，隠れ端末は存在しないが，複数のノードが同時に RTS フレームを送信すると，RTS フレームの衝突が起きると考えられる。その後，ノードは RTS フレームの再送を行うが，衝突が何度も続くと，各ノードは，そのパケットの送信を諦めていまい，パケットを破棄してし

まう。よって，RTS/CTS を利用すると，RTS/CTS フレームによって帯域を消費してしまい，パケットを損失してしまうと思われる。したがって，損失したパケットを補うために，再送機能を持つ確認応答機能は有効であるといえる。

確認応答機能を使用せず RTS/CTS も使用しない場合の到達率は 1 ホップ確認応答機能よりも低くなっている。また，パケットサイズを大きくすると，その到達率はさらに低下している。これは，確認応答機能を使用せず RTS/CTS も使用しない通信では，パケットの損失が多く，エンドツーエンド通信における信頼性は低くなることが分かる。一方，RTS/CTS を使用しない 1 ホップ確認応答機能の場合，パケットサイズを大きくしても，高い到達率を示している。よって，RTS/CTS を利用しない場合でも，確認応答機能を使用すると信頼性は向上している。

表 1 ：パケット到達率（%）の計測結果

	確認応答なし	EtoE 確認応答	確認応答なし	1 ホップ確認応答
RTS/CTS	あり	あり	なし	なし
100Byte	98.7	100.0	98.2	99.9
1000Byte	98.7	100.0	81.3	99.7

1.7 まとめ

本稿では，アドホックネットワークテストベッドフレームワークの開発として，ルーティングプロトコルを容易に実装，交換するためのルーティングフレームワークの開発と，通信実験によるルーティングフレームワークの動作確認について述べた。また，エンドツーエンド通信の信頼性向上のための確認応答機能の開発と，その性能評価について述べた。

しかしながら，本実験規模はまだ小さく，通信を行っているノード対が 1 つしか存在せず，全てのノードは移動しないという単純なネットワークである。また，通信実験を屋内で行ったため，電波の拡散などの影響がほとんどない。そのため，実機におけるアドホックネットワークの問題点が現れた結果ではなかった。

今後の課題としては，まず別々に開発されたこの 2 つの機能を 1 つのソフトウェア上に実装し，動作確認を行うことが挙げられる。また，実験規模を大きくし，通信を複雑化した状況で通信実験を行い，修正や改良を行うことや，そうした実験結果から得られた実機におけるアドホックネットワークの問題点について検証していくことなどが挙げられる。

2 アドホックネットワークにおけるサービス提供グループ動的構成によるサービス発見方式

2.1 はじめに

アドホックネットワーク[1]とは，交換機や基地局に依存しない，交換機能を持つモバイル端末だけで構成される自律分散型ネットワークのことを言う。アドホックネットワークでは，端末自体が交換機機能を有しているため自由に動かすことができる。遠くに離れたユーザの通信は，相手のユーザの途中にいるユーザの端末を経由することにより行うことができる。一般にネットワークを用いるアプリケーションは，他のノード上で提供されるサービスの利用を必要とするクライアントとしてモデル化することができる[6]。アドホックネットワークでは固定された施設で構成した固定ネットワークのように管理を行うことは困難となるため，クライアントはどのようなサービスがネットワーク上で提供されており，どのノードがそのサービスを提供しているかをあらかじめ把握することはできない。そのため，クライアントが要求するサービスを自動的に発見するサービス発見方式は，サービスを利用するための煩わしい設定の手間からユーザを解放するとともに，サービスの迅速な利用開始を可能とするために必要となる。

サービス発見方式として，SLP（Service Location Protocol）[7]，Salutaion[8]，Jini[9]などは基本的にネットワーク上で提供されているサービスの，たとえば IP アドレス，ポート番号，および設定パラメータなどから構成情報を集めたセンタサーバをあらかじめ設け，クライアントがセンタサーバに問い合わせることでサービスの発見

を行う。この固定的に設置されるセンタサーバを用いる方式は一般にアドホックネットワークに適用することは困難となる[10, 11]。そこでセンタサーバを用いずに，クライアントがサービスの発見要求をブロードキャストしサービスを提供するノードが応答する方式が提案されているが，サービス発見要求を行うクライアント数の増加に伴いブロードキャストするメッセージ数が増大し輻輳の原因となってしまう問題があった。そこでサービス発見要求に対する応答メッセージを中継するノードがメッセージ内の構成情報をキャッシュし，あらたなサービス発見要求を受信した場合にキャッシュを元に応答することでブロードキャストするメッセージ数を削減する方式が提案されている[12]。しかしながら，メッセージ数の削減を図る一方で本来発見されるべきサービスを発見できなくなってしまう問題があった。そこで本研究では，従来方式よりもサービス発見要求を行うためのブロードキャストするメッセージ数を削減することを目的として，アドホックネットワークにおけるサービス発見方式を提案する。

2.2 サービス発見方式の提案

2.2.1 提案方式によるサービス発見

提案方式の最大の特徴は，ユーザからのサービス発見要求によるサービス提供グループ動的構成である。既存方式（**図10**(a)）では，ユーザがサービス種別を指定しサービス発見要求を行う Service Request メッセージをブロードキャストする。Service Request を受信したノードが，受信した Service Request に指定されるサービス種別と一致するサービスを提供していない場合，受信した Service Request をさらにブロードキャストする。この Service Request の受信と再ブロードキャストは Service Request に指定されたサービス種別と一致するサービスを提供するノードが Service Request を受信するまで行われる。Service Request に指定されたサービス種別と一致するサービスを提供するノード（サービスプロバイダ）が Service Request を受信した場合，提供しているサービスの構成情報を含む Service Reply メッセージをユーザに対して返信する。既存方式（**図10**(a)）では，ユーザのサービス要求A，B，Cに対して，サービスプロバイダ SP（A），SP（B），SB（C）のそれぞれがユーザからのサービス発見要求を受信し，ユーザへ Service Reply を返信している。提案方式（**図10**(b)）でも既存方式と同様にユーザがサービス種別を指定しサービス発見要求を行う Service Request メッセージをブロードキャストする。提案方式では，サービスプロバイダがユーザからのサービス発見要求に応じて，サービス提供グループを動的に構成する。ここで，サービス提供グループのサイズ，グループのメンバ数をあらかじめ指定しておく必要がある。結果として，他のユーザが同じサービスを要求するために Service Request メッセージをブロードキャストするとき，すでに構成されたサービス提供グループのメンバの一つがこのメッセージを受信することにより，サービス発見のためのブロードキャストメッセージを削減することできる。提案方式（**図10**(b)）では，ユーザのサービス発見要求A，B，Cに対して，サービスプロバイダ SP（A），SP（B），SB（C）がサービス提供グループを構成していれば，サービス要求を受信したサービスプロバイダ SP（A），SP（B），SB（C）のうちの一つがユーザからのサービス発見要求を受信するだけで，ユーザへ Service Reply を返信する。

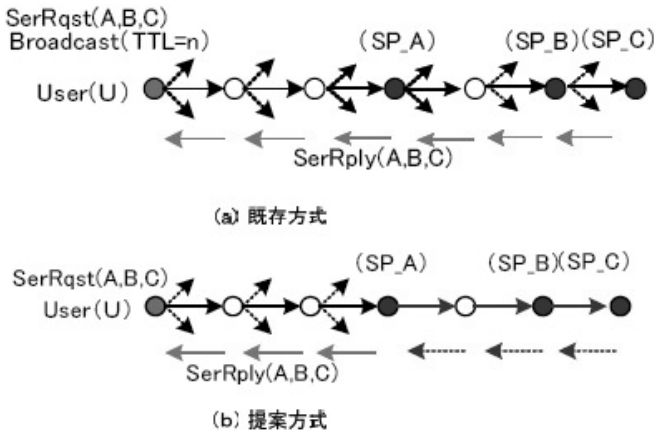

図10：既存方式と提案方式によるサービス発見

ユーザのサービス発見要求のときに，すべてのサービスを探す必要がないため，サービス発見時のブロードキャストメッセージの削減を実現することができる。これは，アドホックネットワークの大規模化に伴い，ブロードキャストメッセージの削減効果が大きくなると考えられる。また，既存方式より短時間でサービスを発見することができる。具体例（**図11**）を用いてサービス提供グループの効果を示す。ユーザのサービス発見要求によって，サービスａ，ｂ，ｃ，ｄ，ｅの五種類のサービスによって構成されているサービス提供グループGが構成されているとする。ユーザが TTL=n で Service Request（G）をブロードキャストする（**図11**）。この Service Request メッセージを受信したGのメンバノードｄは，この Service Request メッセージを他のサービス提供グループメンバに送信する（**図11**）。各メンバの状態を確認した後に，メンバノードｄがユーザに Service Reply を返信することにより，ユーザはサービスを発見できる（**図11**）。

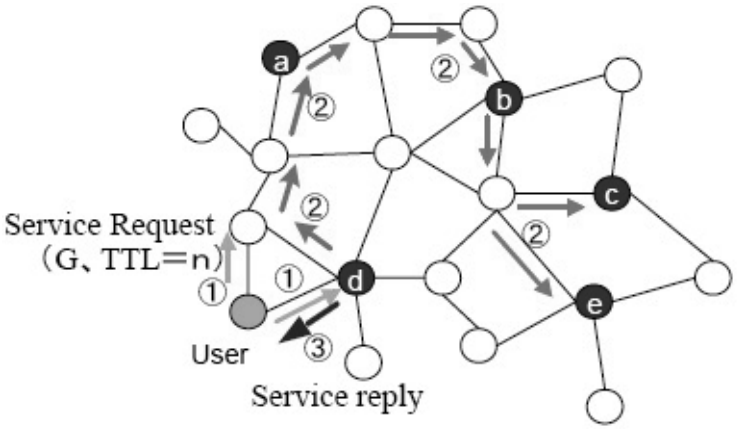

図11：提案方式によるサービスの発見手順

2.2.2　サービス提供グループ動的構成

本節では，アドホックネットワーク上でのサービス提供グループの初期構成と再構成の方法を示す。

サービス提供グループの初期構成　サービスプロバイダ SPSP（1），SP（2），SP（n）がそれぞれサービス Ss（1），s（2），s（n）を提供しているとする。ユーザがサービス発見するために，Service Request S（r）をブロードキャストする。ここで，S（r）=s（i），… s（j），… s（k）とする。S（r）の構成要素である s（i）サービスを提供している SP（i）が同じ Service Request s（r）を複数のユーザから d 回受信した場合，動的にサービス提供グループを初期構成する。その手順を以下に示す。

1．SP（i）は Invite メッセージをブロードキャストする。

2．Invite メッセージを受信した S（r）のサービスを提供しているサービスプロバイダは Join メッセージを SP（i）に返信する。

3．SP（i）がすべてのメンバから返信を受け取って，サービス提供グループを構成する。このとき，グループメンバ間の経路を MAODV[13]に代表されるマルチキャストルーティングプロトコルの手順に従いツリーを構成する。

サービス提供グループ再構成　ユーザのサービス発見要求に対して柔軟に対応するため，サービス提供グループに対して新たなサービスの追加，およびサービス削除などを行う必要がある。ユーザのサービス発見要求に応じて,サービス提供グループ内のサービスの追加と削除を行うことにより,サービス提供グループの再構成を行う。

⑴　サービス提供グループのサービスの追加：アドホックネットワーク上でサービス提供グループ G（a，b，c，d，e）が存在する。ユーザが新たなサービス（G+f）を要求すると，サービス提供グループ G はこの要求に対応する必要がある。追加の手順は以下の通りである。ユーザが新たな Service Request（G+f）をブロードキャストする（**図12**）。サービス提供グループ G のメンバノード d がこの Service Request メッセージを受信したときに，サービス提供グループ G には，サービス f が含まれてないことを認識する。このとき，d は Invite（f）メッセージをブロードキャストする（**図12**）。サービス f を提供するノード f が Invite（f）メッセージを受信した場合，ノード f はサービス提供グループ G に参加するために Join（G）メッセージを TTL<

H（d，f）でブロードキャストする（**図12**）。ここで，H（d，f）はノード d とノード f 間のホップ数とする。この処理によって，f に一番近いサービス提供グループ G のメンバノードに対して Join メッセージを返信する（**図12**）。

⑵ サービスの削除：サービス提供グループのメンバ b がサービスを提供できなくなる，また，ユーザからのサービス発見要求に含まれなくなった場合，b を削除することにより，サービス提供グループ G を再構成する必要がある。この処理は以下の通りである。まず，メンバノード b は Leave メッセージを隣接のグループメンバに送信する（**図13**）。Leave メッセージを受信したノード a は，その情報に基づいてノード c と e に対してのみ，invite（）メッセージを送ることによって，局所的にツリーを再構成する（**図13**）。

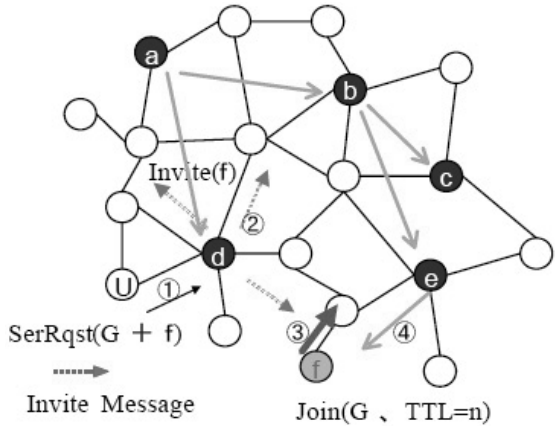

図12：グループメンバの追加

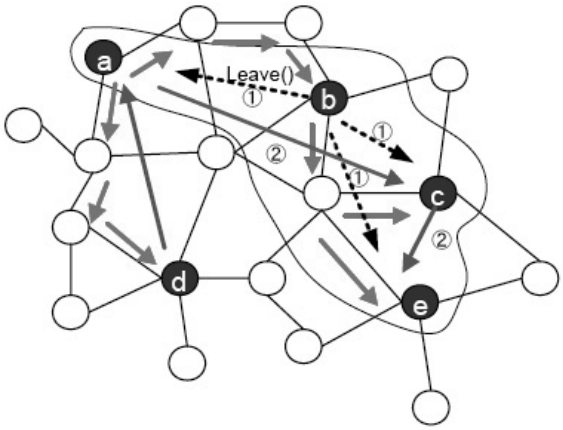

図13：グループメンバの削除

2.2.3 サービス提供グループの消滅

サービス提供グループは生存時間を持っているため，ある時間内利用されない場合，サービス提供グループは自動的に消滅する。

2.3 シミュレーション評価

提案方式の有効性を確認するために，NS ネットワークシミュレータ[14]を用いて評価を行う。各ノードの電波の到達距離は 250M とし通信速度を 2Mbps とする。クライアントの移動モデルは，ランダムウェイポイントモデルとする。各ノードは 1500M×300M の領域をランダムに選択した位置から移動先の位置をランダムに選択し，あらかじめ指定する速さの中かランダムに選択した速さで移動する。評価結果は各条件で10回のシミュレーションを試行した結果の平均値とする。

2.3.1　評価結果

サービス発見率の評価結果　**図14**では，サービス数10，ユーザ数１から20の提案方式とブロードキャスト方式のサービス発見率を示した。ここで，サービス発見率は Service Request メッセージ総数に対する Service Reply メッセージ総数の割合とする。**図14**に示すとおり，ブロードキャスト方式では，提案方式に比べ，サービス発見率が著しい低く，ユーザ数の増加に従いサービス発見率が減少してしまっている。これに対して，提案方式はブロードキャスト方式に比べ，ユーザ数が増加しても発見率の大幅の減少はなかった。

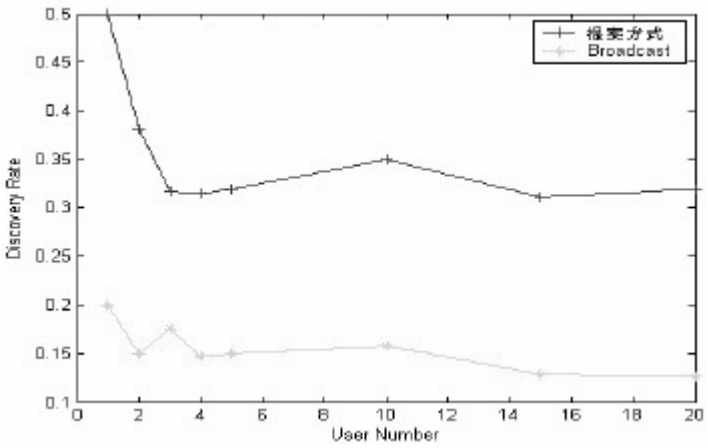

図14：サービス発見率のシミュレーション結果

応答時間の評価結果　**図15**(a)に示すとおり，ユーザ数を固定する時，サービス提供グループのサービスの増加により，サービス発見の応答時間は，大きく変化することはなかった。**図15**(b)に示すとおり，サービスを数固定する時，ユーザ数の増加に伴い，応答時間は，ゆるやかに増大することが分かった。

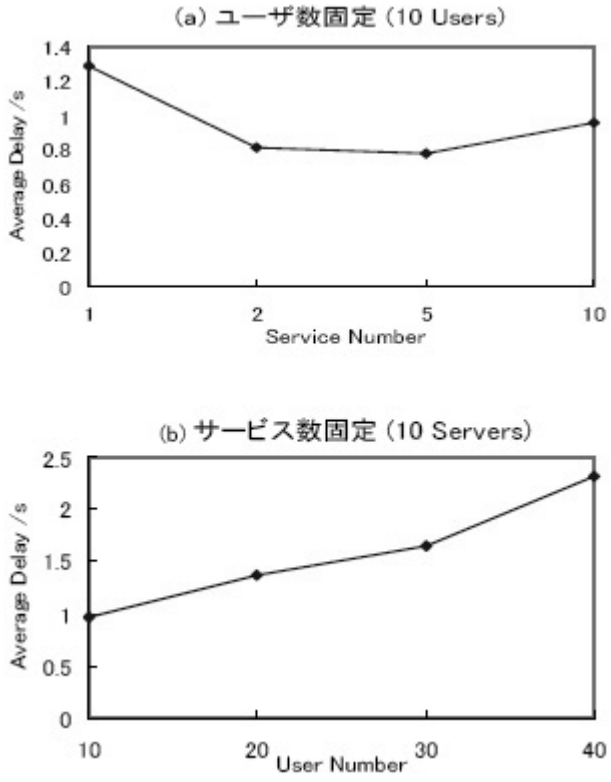

図15：応答時間のシミュレーション結果

2.3.2 考察

従来方式では，サービス発見要求を行うクライアント数の増加に伴いブロードキャストするメッセージ数が増大し輻輳の原因となってしまう問題があった。シミュレーション結果によって提案方式は従来方式に比べ，サービス発見率は大きく，提案方式の有効性を示すことできる。また，提案方式では，応答時間は大きく変動することはなかった。

2.4 まとめ

本稿では，メッセージ数を削減するために，サービス提供グループ動的構成による，アドホックネットワークにおけるサービス発見方式を提案した。今後，メッセージ数等の詳細な評価を行う予定である。

参考文献

[1] 角田，大田：アドホックネットワークルーティング，オペレーションズ・リサーチ，Vol.48 No.3，pp.189-195（2003）.

[2] 小牧：無線 LAN とユビキタスネットワーク，丸善株式会社，2004.

[3] C.E.Perkins and E.M.Royer: Ad hoc on-demand distance vector routing, 2nd IEEE Workshop on Mobile Computing Systems and Applications, pp.90-100, 1999.

[4] 田坂：情報ネットワークの基礎，数理工学社，2003.

[5] 村田：マルチメディア情報ネットワーク，共立出版株式会社，1999.

[6] E. Guttman, et al.: Service Location protocol, Version 2, IETF, RFC 2165, 1999.

[7] E. Guttman, C. Perkins, J. Veizades, and M. Day: Service Location Protocol, Version 2, IETF RFC 2608, June 1999. http://www.ietf.org/rfc/rfc2608.txt.

[8] The Salutation Consortium, Salutation Architecture Specification, www.salutation.org.

[9] Sun Microsystems, JINI technology, www.jini.org.

[10] M. Nidd: Service Discovery in DEAPspace, IEEE Personal Communications, August 2001.

[11] L. M. Feeney, B. Ahlgren, and Assar Westerlund. Spontaneous networking: An application-oriented approach to ad hoc networking. IEEE Communication Magazine, 39(6), 2001.

[12] B. Raman, P. Bhagwat, S. Seshan: Arguments for cross-layer optimizations in Bluetooth scatternets, Symposium on Applications and the Internet, pp.176-184, 2001.

[13] E. Royer, C. Perkins: Multicast Ad hoc On-Demand Distance Vector (MAODV) Routing, Internet-draft, July 2000, draft-ietf-manet-maodv-00.txt.

[14] USB/LBNL/VINT Network Simulator ns (Ver.2),http://www.isi.edu/nsnam/ns.

〈発 表 資 料〉

題　　名	掲 載 誌 ・ 学 会 名 等	発 表 年 月
Development of Testbed Framework for Ad Hoc Networks	Proc. Wireless Personal Multimedia Communications (WPMC'05)	September 2005
Dynamic Load Balancing Using Network Transferable Computer	Proc. 25th IEEE International Conference on Distributed Computing Systems Workshops	June 2005
Effective division and merger of the autonomous clustering scheme for highly mobile large ad hoc networks	Proc. 2005 IEEE International Symposium on Circuits and Systems (ISCAS2005)	May 2005
A new multicast routing protocol based on autonomous clustering in ad hoc networks	Proc. 7th IEEE International Symposium on Autonomous Decentralized Systems (ISADS2005)	April 2005
An improved autonomous clustering scheme for highly mobile large ad hoc networks	Proc. 7th IEEE International Symposium on Autonomous Decentralized Systems (AHSP2005)	April 2005
A dynamic index allocation scheme for peer-to-peer networks	Proc. 7th IEEE International Symposium on Autonomous Decentralized Systems (AHSP2005)	April 2005
A multiagent-based path rerouting method in MPLS networks	Proc. 7th IEEE International Symposium on Autonomous Decentralized Systems (WCCIA2005)	April 2005
A class of hierarchical routing protocols based on autonomous clustering for large mobile ad hoc networks	IEICE Transactions on Communications (Special Issue on Networking Technologies for Mobile Internet Systems)	September 2004
エリア分割によるモバイルエージェントへのメッセージ伝達法	電子情報通信学会和文論文誌 B	June 2004
An adaptive automobile control system using scheduling by imprecise computation and multiagent-based traffic information exchange and its experimental evaluation	Proc. 24th IEEE International Conference on Distributed Computing Systems Workshops (ADSN2004)	March 2004