モバイルコンピューティングと
ワイヤレス通信 16－5
（2001．2．14）

より信頼性を考慮したアドホックマルチホップルーチングの検討

奥田 隆弘† 井野 宇大‡ 石原 進‡ 渡辺 尚‡

近年アドホックネットワークの要求が高まっており、各種のプロトコルが提案されている。アドホックネットワークの中でも、対象となるノードが広範囲に分散しているようなネットワークではアドホックマルチホップルーチングが用いられる。ITS における車々間通信のようにノードが高速に動作する場合にも、より高い信頼性を確保する必要がある。本稿で提案する Mutihopping OVErlapped NETworks(MOVENET)は、各ノードごとに定義するネットワークをオーバラップさせる構成と、経路範囲設定をしたフラッディングベースによるパケット転送によって、高速トポロジ変化時においてもより高い信頼性の確保を目指している。

A Reliable Study of Ad Hoc Multihop Routing

Takahiro Okuda†, Udai Ino‡, Susumu Ishihara‡, Takashi Watanabe‡

The request of the ad hoc network is rising in recent years and a various protocol is proposed. In the ad hoc network, when nodes are widely distributed, ad hoc multihop routing is used. Most of research ignore the reliability, however, when the node works at high speed like the inter-vehicle communication in ITS, the higher reliability must be achieved. Mutihopping OVErlapped NETworks(MOVENET) is to built more reliable routing. The main ideas composing an ad hoc network of a sub network defined by each node, the sub network make wrap each other and flooding base packet transfer with setting a route zone.

1．はじめに

情報端末や無線通信装置の小型軽量化にともない、無線通信機能付きの端末を持ち歩いて作業するモバイルコンピューティングが普及しはじめている。

その典型的な例はノート PC と携帯電話をつないでネットワークに接続する方法である。この方法は既存のネットワーク技術を利用して実現できるが、キャリアを介するために基地局の状態や課金の問題が発生する。

一方で携帯端末が無線通信装置を持つならば、自分の近隣の端末とはキャリアを介さずに直接通信した方が効率的であると考えられる。そのように周辺に集まったノード同士で一時的に構成されるネットワークはアドホックネットワークと呼ばれている。

会議室などに集まった端末同士でアドホックネットワークを構成するには、Bluetooth[1]や IrDA[2]といった規格が提案され実用化が進められている。これらの規格は限られた狭い範囲でネットワークトポロジの変化があまりない状況での利用には適している。

ところが、近年研究開発が進められている ITS（Intelligent Transport Systems＝高度道路交通システム）における車々間通信などを想定した場合、それらの技術では対応しきれない。近隣の車同士で構成されるネットワークはどこからどこまでが一つのネットワークであると定義するのが非常に難しく、また日本全国に存在する全ての車で一つのネットワークを構築するのはおよそ現実的ではない。さらに車はそれぞれが独立して移動するため、ネットワークトポロジは絶えず高速に変化している。

本稿では特に車々間通信において必要となる高速トポロジ変化への対応と信頼性の向上を目指したアドホックマルチホップルーチングプロトコルを提案する。以降、2章でアドホックマルチホップルーチングの概要と関連研究を述べ、3章ではより信頼性を考慮したアドホックマルチホップルーチング方式を提案する。

† 静岡大学大学院情報学研究科
Graduate School of Information, Shizuoka University
‡静岡大学情報学部
Faculty of Information, Shizuoka University

そして４章でまとめとする。

２．アドホックマルチホップルーチング

近距離で閉じたグループ内で構成されるアドホックネットワークならば、全てのノードが互いに無線通信範囲内に存在することを前提にできる。その場合はルーチングを行う必要がない。ところが無線通信の到達距離に比べて遥かに広範囲に分散しているノード群によって構成されるアドホックネットワークの場合は、複数のノードを中継して目的のノードにパケットを転送する方法が考えられる。それがアドホックマルチホップルーチングである。図１のような例の場合、ノードAB 間とノード BC 間は互いに双方向通信ができるが、ノード AC 間は直接には通信ができない。そこでBを中継ノードとすることでノードＡとノードＣの間でも通信を可能にする。

マルチホップルーチングによるネットワークでは、あるノードから他のノードにパケットを転送する経路は複数のノードを中継する複数の無線リンクによって構成される。本稿では以下の用語を用いる。

Source
　パケットを生成した最初のノード
Destination
　マルチホップして届くパケットの最終目的ノード
Sender
　ある無線リンクにおける送信ノード
Receiver
　ある無線リンクにおける受信ノード

アドホックマルチホップルーチングプロトコルの例としては以下のものが挙げられる。しかしこれらのプロトコルではトポロジが高速に変化するような状況に十分に対応できているとは言えない。

・DSR[3]

Source は Destination までの経路を発見するために、Route Request パケットをフラッディングする。これを受信した Destination は Route Reply を Source に返す。この Route Request/Route Reply パケットはそれ自身が通過した全ての経路を記憶しているため、ノードは Destination までの完全な経路を知ることができる。Source は Destination までの全ての経路を指定してパケットを送信する。

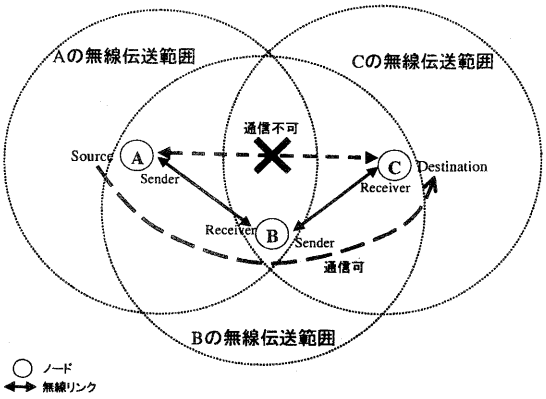

図１．マルチホップルーチング

・AODV[4]

経路の発見には DSR と同じ Route Request/ Route Reply を用いるが、通過した経路を全て記録するのではない。各中継ノードが Destination に到達するためには次にどのノードに送れば良いかを記録しておく。したがって Destination 宛のデータパケットには経路情報は含まれず、各中継ノードで上の経路表に従って転送される。

・ZRP[5]

各ノードは周囲のノードと定期的に制御メッセージを交換することで自ノードから n ホップ先までのトポロジの状況を把握する。そして宛先までの経路が分かっている場合はその経路に向けてパケットを転送し、経路の分からないパケットは自分から $n+1$ ホップ目に存在する全てのノードに到達するように転送する。

３．マルチホップオーバラップネットワーク

3.1.ネットワーク構成

本稿で提案するアドホックマルチホップルーチングプロトコルでは、各ノードが自ノードから２ホップ先までの双方向の無線リンクトポロジを認識する。この自ノードから２ホップ以内のネットワークを自ネットワークと定義する。そのためノードのごとに異なるネットワークが存在し、それぞれのネットワークをオーバラップさせて全体のネットワークを構成する。そして、オーバラップした部分のノードがパケットを中継することにより自ネットワーク外のノードとの通信

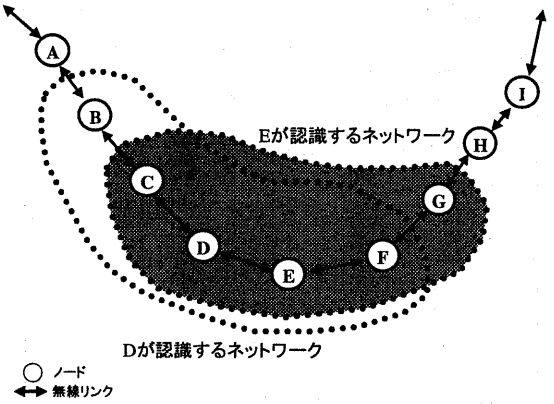

図2．オーバラップネットワーク

を可能にする。

図2の例において、ノードDはノードB～Fまでを自ネットワークと認識し、ノードEはノードC～Gを自ネットワークと認識している。つまりノードC～Fがオーバラップしているのである。

後で述べる選択的フラッディングを行うためには、1ホップ先の情報だけでは足りない。またトポロジ認識範囲を広げると制御メッセージ量が増大し、トポロジ情報の伝播にも時間がかかる。これらの理由により、2ホップ先までを自ネットワークと定義する。

局所的に自ネットワークを定義して互いにオーバラップさせる特徴により、本稿で提案するアドホックマルチホップルーチングプロトコルを MOVENET（Multihopping OVErlapped Networks）と呼ぶ。

3.2. 2ホップトポロジデータベース

MOVENETでは各ノードがHELLO messageと呼ばれる制御パケットを定期的にブロードキャストすることにより、ノードが互いに自ネットワークを認識できる。各ノードは1ホップ先のノードからのHELLO messageを受信して最近の周辺トポロジ状態を把握する。したがって、HELLO messageの交換間隔を短くするとトポロジ変化への追従能力が高くなるが制御用バンド幅が増大し、逆に交換間隔を長くすると制御用バンド幅は減少するがトポロジ変化への追従能力が低くなる。

・Neighbor Table

Neighbor Tableは各ノードによって保持され、自ノードから2ホップ先までのトポロジの情報が記憶されている。Neighbor Tableの各エントリは対応するノードからのHELLO messageを受信するごとに更新される。一定期間以上HELLO messageが受信されなくなったエントリは、自ノードの近隣にいなくなったものとしてエントリを削除する。図3にNeighbor Tableのフォーマットを示す。Neighborは自ノードから直接通信できる近隣ノードであり、Next Neighborは対応するNeighborを介して2ホップで到達できるノードである。双方向リンクはその経路で互いに通信可能なことを示し、片方向リンクは該当ノードから自ノードの方向への通信のみが可能であることを示す。

・Packet Type
　パケット識別子 HELLO
・Source
　自ノードのID
・Neighbor[1]～Neighbor[m]
　Neighbor Tableのn[1]～n[m]のIDとリンク状態

図4．HELLO messageパケット

Neighbor	Next Neighbor	リンク状態
n[1]	—	双or片方向
	nn[1]	双or片方向
	：	
	nn[s]	双or片方向
：		
n[m]	—	双or片方向
	nn[1]	双or片方向
	：	
	nn[t]	双or片方向

図3．Neighbor Table

・HELLO message

HELLO messageは自ノードから1ホップ先のトポロジの状態を近隣のノードに通知する。図4のようにNeighbor TableのNeighborとそのリンク状態を列記したHELLO messageを定期的にブロードキャストする。

HELLO messageを受信したノードは自身のNeighbor Tableの対応するエントリを削除し、Neighbor[1]～Neighbor [m]をNext Neighborとするエントリに更新する。この際、そのNext Neighbor内で自ノードがエントリされている場合はSourceノードまで双方向リンクとし、Next Neighborに自ノード

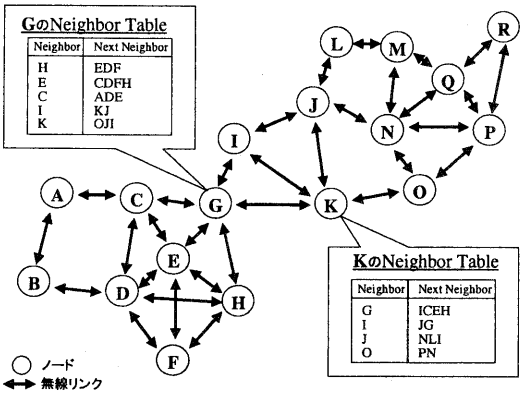

図5．Neighbor Table例

がエントリされていない場合は片方向リンクとする。

図5はHELLO messageの交換が行われた十分な回数の後のNeighbor Tableの構成例である。

3.3.選択的フラッディング

MOVENETにおけるパケット転送は基本的にフラッディングベースで行われる。ただし単純なブロードキャストをするのではなく、ノードがパケットを転送する際に、パケットヘッダでそのパケットを受信すべき宛先ノードを指定する方式で行われる。この方式ではSenderが1つあるいは複数のReceiverを選択するので筆者らはこれを選択的フラッディングと呼んでいる。MOVENETで用いられるデータパケットのフォーマットを図6に示す。

選択的フラッディングでは、各ノードは受信したパケットごとに、以下の2つのルールに従ってそのパケットをフラッディングしない近隣ノード（宛先除外ノード）を判定し記憶する。

(1) 1ホップ前のSenderとそのReceiverには送る必要がない。
(2) 同じシリアル番号のパケットを近隣のノードが送信したら、そのSenderとReceiverには送る必要がない。

宛先除外ノードの決定方法を図7に示す。SNは自身のNeighbor TableでsエントリのNext Neighborに相当する。転送されるパケットはMAC層で送信される順番を待つため、送信待ちキューに入れられる。

- Packet Type
  パケット識別子 DATA / RREQ / RREP
- Distance Count
  Sourceへの遠さを表す値
- Route Zone
  経路範囲の広さを表す値
- Source
  パケットのSourceのID
- Destination
  パケットのDestinationのID
- Serial Number
  パケットのシリアル番号
- Sender
  自ノードのID
- Receiver[1]～Receiver[n]
  次の無線リンクで受信するノードのID
- Data
  データ部分

図6．パケット形式

ノードは周囲の送信状況を常にモニタし、送信待ちキュー内にあるパケットと同じパケットが隣接ノードから送信されることを検知する。検知された場合、該当するパケットの宛先除外ノードを追加していく。

そしてパケットを送信する直前に宛先除外ノードとNeighbor Tableと照らし合わせることにより、必要なReceiver[1]～Receiver[n]をリストアップしたパケットヘッダを生成する。その方法を図8に示す。

またこの動作の具体例を図9に示す。パケットはSourceからDestinationに向かって流されるものとする。ノードAは転送されてきたパケットの次のホップのReceiverにBとCを指定して送信する（①）。これを受信したBとCのうち、例えば先にCが次の送信を開始したとすると、CはReceiverにDを指定して送信する（②）。この際、Bも宛先除外ノードをA, Cに指定して送信待ち状態になっているが、Cの送信を聞くことによりDも宛先除外ノードになる。するとBが次に転送すべき宛先がなくなるため、Bは送信を取りやめる。その後Cで転送されたパケットはD, Fと転送される（③,④）。通常のフラッディングの場合は同じパケットを全てのノードが1回は転送しなければならないところ、この方法ではBとEの送信を抑えることができる。もし仮に②の段階でBが先に送信を始めたとしても、CとEの送信を抑えることができる。

```
s ： 受信パケットの Sender ノード
S ： s の Neighbor 集合
E ： 宛先除外ノード集合

  if 新規に受信したパケット then
   E := S ∪ s
  else
   E := E ∪ S ∪ s;
```

図 7．受信除外ノードの記憶

```
n[i] ： Neighbor Table の i 番目のエントリノード
N    ： Neighbor 集合
NN[i] ： n[i]の Next Neighbor 集合
E    ： 宛先除外ノード集合
R    ： 宛先ノード集合

  R := φ;
  for i =1 to SUM(N) do
   if n[i] ∉ E then
     if (NN[i] + E + N) ≠ ER then
       R := R + n[i];

  if R = φ then パケット破棄 else 送信;

※ NUM(X) は集合 X に含まれるノード数
```

図 8．受信ノードの決定

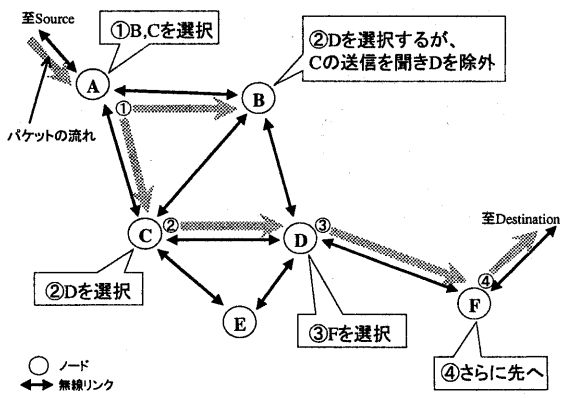

図 9．Receiver 選択の例

```
N    ： Neighbor の集合
S    ： 受信パケットの Sender の Neighbor 集合
d    ： 受信パケットの Distance
offset： 計算される Offset
d'   ：更新される Distance

  offset := NUM(N – SN) – 1;
  d' := d + offset
```

図 10．Distance の計算

3.4. 経路範囲設定

MOVENET ではユニキャストを行う際にも選択的フラッディングによるパケット転送を用いる。しかし単純にフラッディングするだけでは、パケットは Source ノードから四方八方に散乱してしまうため非常に効率が悪くなる。そこで Source ノードから Destination ノードの方向に存在するノードに限定した経路範囲を設定して、その経路範囲内においてフラッディングベースのユニキャストを行う。

また、経路範囲を狭く設定すれば経路数が少なくなって信頼性が低下するがバンド幅消費も少ない。逆に、経路範囲を広く設定すれば複数経路が存在して信頼性が向上する代わりにバンド幅消費も増大する。

・Distance Count

一般にマルチホップルーチングでは、ノード間の距離を表す値としてホップカウントが用いられる。しかし MOVENET では、ある経路において、“その経路上のノードから 1 ホップ以内に存在するノードの合計”の値を用いる。この値が小さい経路の方がパケットの輻輳が軽減できると考えられる。

図 10 に各ノードでの Distance の算出方法を示す。パケットには 1 つ前ホップの Sender が算出した Source までの Distance が書かれているため、自ノードではそのパケットに対する Offset 値を計算し、パケットの Distance 値にインクリメントして次ホップに送信する。特に自ノードから自ノードへは Distance 値が 0 で Offset が Neighbor の数とする。したがってパケットが生成される時の Distance の初期値はそのノードの Neighbor 数を記載する。図 11 で示ように、ホップ数では同じ値になる二つの経路でも、Distance Count では隣接ノードの数が少ない経路の値が小さくなる。

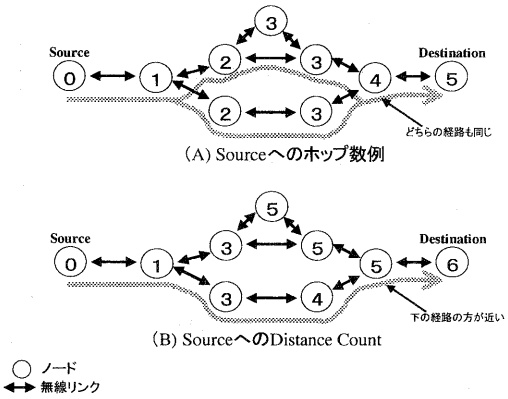

図 11. ホップ数と Distance Count の違い

・Route Table

各ノードは任意のノードまでの Distance Count を Route Table に保持している。図 12 に Route Table のフォーマットを示す。ある Source ノードからのパケットを受け取った際に Route Table 中の Source ノードのエントリを更新する。Distance はパケットに記載されている値で、Offset は前述の方法で受信時に計算する。また経路情報の新しさを知るために Sequence としてパケットのシリアル番号も記録する。

Distance 値の更新方法は以下の 2 つのパターンがある。なお長期間にわたって更新されないエントリは削除すべきである。

(1) より新しいシリアル番号を持ったパケットを受信した時は更新する。
(2) シリアル番号が同じで Distance 値が小さいパケットを受信した時は更新する。

この更新作業は主に Route Request / Route Reply パケットによって行われる。また、通常のデータパケットの転送時でも、データパケットのヘッダ情報から、随時最新の値に更新できる。

・Route Request

初期状態では目的のノードに関する Route Table エントリが存在しないため、ある Destination への経路範囲を設定するために Route Request パケットを周囲に選択的フラッディングする。 この場合 Route Zone 値はパケットが伝播する範囲の TTL 値に相当する。タイムアウト期間を過ぎても対応する Route Reply が返

Node	Sequence	Distance	Offset
Node[1]	Seq[1]	Dist[1]	Offset[1]
		:	
Node[k]	Seq[k]	Dist[k]	Offset[k]

図 12. Route Table

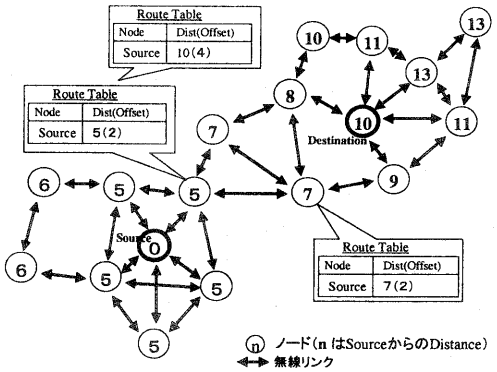

図 13. Route Request 時の例

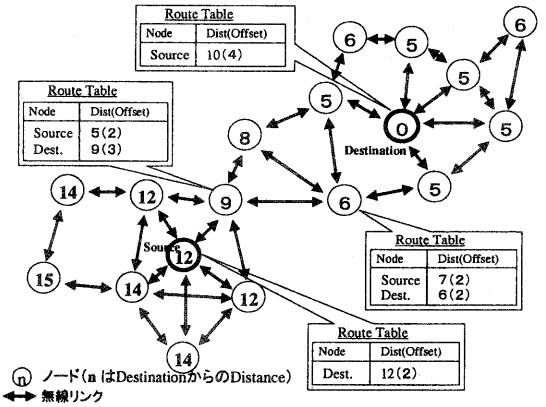

図 14. Route Reply 時の例

ってこない場合は、その Destination には到達できないことになる。

図 13 に Route Request でパケットが転送された場合に各ノードで記録される Route Table の Source ノードからの Distance 値を示す。

・Route Reply

自ノード宛の Route Request パケットを受信したノ

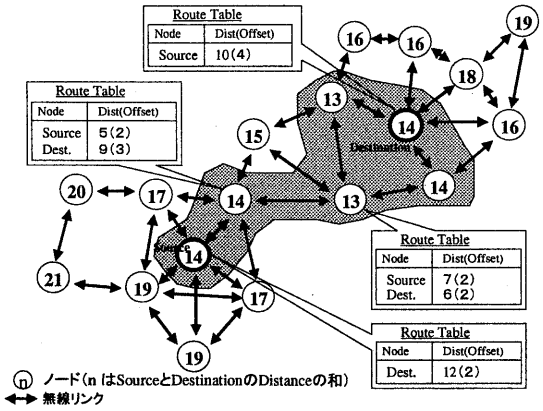

図 15. 経路範囲の選択例

ードは Route Reply パケットを返信する。これも選択的フラッディングによって行う。

図 14 に Route Reply でパケットが流れた場合に各ノードの Route Table で記録される Destination ノードからの Distance 値を示す。

このように Route Request / Route Reply を交換することにより Source と Destination 間およびその周辺に存在する各ノードの Route Table には Source と Destination からの Distance が記録されることになる。

・範囲選択

ある Source からある Destination の経路を考える場合、周辺にある中継ノードとなる可能性のある各ノードでは Route Table に記録された Source への Distance と Destination への Distance の和をとることで、自ノードと Source と Destination の位置関係が認識できる。この和の値は Source と Destination 間の適当な経路において小さな値になる性質がある。

図 6 の Route Zone 値は Distance 値の和が Route Zone 値以下のノードに限って中継を許すことを意味する。したがってパケットを生成する時にこの値を最小限にしたり大きく設定したりすることによりフラッディングする範囲を制御することができる。この最小値は Source ノードが保持する Route Table の Destination エントリの (Distance + Offset) である。Route Zone をこの値とすれば、Source と Destination の間で 1 本以上の経路が存在する。

図 15 では各ノードにおいて Source と Destination について Distance の和をとった値を示している。この場合 Route Zone 値として 14 を設定し、Distance の和が 14 以下の値を持つノードを選び出すと網掛け部分のように 2 本の経路が存在することが分かる。

図 16. パケット中継シーケンス

3.5. ルーチング

中継ノードがデータパケットを転送するシーケンスを図 16 に示す。まず重複受信をチェックした後、受信されたパケットの Source と Distance の情報で Route Table を更新する。そして自ノード宛のパケットならば、データを上位層に渡したり Route Reply を返信するなどの処理を行う。パケットが自ノード宛でない場合は次に Route Zone 値を調べる。もし Route Table をもとに計算した値が Route Zone よりも大きければそのパケットを破棄する。そして宛先除外ノードを設定して送信待ちキューに投入する。前述のように、ノードは隣接ノードのパケット送信をモニタして送信待ちキュー内のパケットの宛先除外ノードを追加する。最後に実際の宛先ノードを決定してパケットの送信を行う。

各中継ノードがこのような処理を行うことにより、パケットは Source から Destination への指向性を持ったフラッディングにより転送される。

4. おわりに

本稿では ITS における車々間通信にも対応できるアドホックマルチホップルーチング方式として

MOVENETを提案した。MOVENETは自ノードから2ホップ先までを自分のネットワークと定義して認識することにより、ノードごとのネットワークをオーバラップさせることで全体のネットワークを構成する。そして、経路範囲を設定したフラッディングベースのパケット転送を行うことにより、ノードの高速トポロジ変化状況において、より高いトポロジ変化追従能力とより信頼のあるパケット転送能力を持つ。

本稿ではMOVENETプロトコルの基本的な仕様提案のみを行った。今後は実験により性能の評価を行う予定である。

参考文献

1) 相原達, "近距離無線通信規格 Bluetooth", bit 2000.10, 共立出版, pp.8-16, October 2000

2) 片山三千太, "赤外線通信規格 IrDA", bit 2000.10, 共立出版, pp.2-7, October 2000

3) draft-ietf-manet-dsr-03.txt, INTERNET DRAFT, October 1999

4) draft-ietf-manet-aodv-06.txt, INTERNET DRAFT, July 2000

5) draft-ietf-manet-zone-zrp-02.txt, INTERNET DRAFT, June 1999

6) draft-ietf-manet-cbrp-spec-01.txt, INTERNET DRAFT, August 1999

7) 奥田隆弘, 渡辺尚, "車々間通信のためのマルチホップオーバラップネットワークの検討", 電子情報通信学会基礎・境界ソサイエティ大会, A-17-28, 2000