社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2007－MBL－43 (5)
2007－ITS－31 (5)
2007／11／21

リアクティブ型アドホックルーティングにおけるフラッディング自己最適化手法

藤田 祥[†1] 安本直史[†1] 江崎 浩[†1]

無線アドホックネットワークではこれまで様々なルーティングプロトコルが研究されてきた．その中でもリアクティブ型プロトコルでは通信の開始に必要な経路のみを確立するため，定期メッセージの交換は行なわず，通信が少ない場合のオーバーヘッドが少ない．その一方でノードが密に配置された場合や通信頻度が高い場合にはオーバーヘッドが大きくネットワーク全体が不安定になってしまう．本研究ではリアクティブ型プロトコルにおいて近隣ノードの情報を既存のメッセージに埋め込むことで定期メッセージを用いずに近隣ノードの情報を集め，それによりメッセージのフラッディングを効率化する．フラッディングは繰り返される度に自己最適化することになる．プロトタイプシステムをリアクティブ型プロトコル DYMO の機能拡張として実装し，Qualnet ネットワークシミュレータ上で性能評価を行った．結果，経路の接続成功率を保ちながらルーティングメッセージ数が最大で 17%削減出来ることを確認した．MAC 層の競合オーバーヘッドまで考慮するとネットワークのスループットと安定性に大きく寄与すると考えられる．

Self-Optimization of Flooding on Reactive Ad-Hoc Routing

Sho Fujita,[†1] Tadashi Yasumoto[†1] and Hiroshi Esaki[†1]

A variety of routing protocols for mobile ad-hoc networks have been studied. Among them reactive routing protocols are efficient in scenarios where only small sets of nodes are communicating, since they establish and maintain only routes to be used. On the other hand, they are not for scenarios where nodes are arranged closely and where bursty communications occur. That is because they establish routes one by one and especially because they adopt the traditional flooding algorithm that can produce many redundant messages. In this paper, we propose a self-optimized flooding algorithm for ad-hoc reactive routing protocols. Each node gathers information of neighbor nodes within two hops without exchanging periodic messages, then utilize them to suppress rebroadcasts of received messages. We applied the proposed algorithm to DYMO and evaluated its effectiveness through Qualnet Network Simulator. The result showed it suppressed at most 17% of message rebroadcasts without compromising success rates to establish new connections.

1. はじめに

DSR[8], AODV[10], DYMO[2] は必要な経路のみを必要な時に解決・管理するリアクティブ型のアドホックルーティングプロトコルである．ネットワーク内の全てのノードへの経路を最新に保つプロアクティブ型のプロトコルと比較して通信が少ない場合にはルーティングのオーバーヘッドを低く抑えることが出来る．一方，確立された経路あたりのオーバーヘッドについては，1回のルーティングメッセージ交換で経路を1組ずつ確立していることやメッセージの配送に全てのノードが1度ずつメッセージの複製を送信する古典的なフラッディングに頼っていることから大きくなっている．特にノードが密に配置された場合や通信の開始頻度が高い場合には大量のルーティングメッセージが交換され，ネットワーク全体のスループットや安定性に深刻な影響を与えてしまう．したがって，リアクティブ型のルーティング方式で安定したネットワークシステムを構築するためにはノードの配置や通信パターンに制約を課さなければならず，結果としてその適用可能範囲が狭められている．

本研究ではリアクティブ型のアドホックルーティングのフラッディングを効率化させる新しい手法を提案する．まずルーティングメッセージに近隣ノードの情報を付加し，定期メッセージの交換をせずに，フラッディングの過程で各ノードが近隣ノードの情報を収集する．次に得られた近隣ノードの情報を利用して自分が再送しなくても全ての近隣ノードが受信済みであることを期待出来る場合には再送を抑制してフラッディングを効率化する．これを繰り返すことでフラッディングは自己最適化されていく．本手法を適用することでリアクティブ型のアドホックルーティングプロトコ

†1 東京大学大学院情報理工学系研究科
Graduate School of Information and Technology, The University of Tokyo

ルはその定常時のオーバーヘッドを低く抑えるという長所を残したまま，突発的に多数の通信開始を生じた場合にもネットワークを安定に保つことが出来るようになる．

本稿の構成は以下の通りである．まず2節で現在のリアクティブ型アドホックルーティングの問題点について説明を行ない，3節でリアクティブ型のアドホックルーティングのためにフラッディングの自己最適化手法を提案する．4節では提案手法をネットワークシミュレータ上に実装し，その結果を元に提案手法の評価を行なう．5節で関連研究を紹介し，と6節でまとめと今後の展望を述べる．

2. リアクティブ型アドホックルーティング

無線アドホックネットワークとは無線インターフェースを持つモバイルノード（以下ノード）のみで構成されるネットワークである．2つのノードが直接通信出来ない場合にも他のノードにパケットを中継させて接続性を確保する．アドホックネットワークのルーティングとは通信するノードの間の中継ノードを決定する仕組みであり，従来のルーティングと区別してアドホックルーティングと呼ばれる．

その中でもリアクティブ型のアドホックルーティングとはオンデマンド型とも呼ばれ，通信に必要な経路のみを通信開始時に確立・管理する方式である．通信の開始や経路の破壊などのイベントに反応して動作するので，定期メッセージ交換がなく，通信がない場合のオーバヘッドは全くない．本研究の議論は全てのリアクティブ型のアドホックルーティングプロトコルに適用出来るはずだが，具体例が必要な場合にはIETFで提案されているDYMOをもとに説明する．

DYMOによって通信を開始する2ノード間に経路が確立されるまでの基本的な処理は以下の通りである．

(1) 通信開始時に宛先ノードへの経路がなかった場合，開始ノードは宛先ノードに対する経路要求(以下RREQ)メッセージをブロードキャストする．RREQメッセージを受け取ったノードは再びブロードキャストで中継しメッセージは徐々にネットワーク全体に広がっていく．各ノードは中継の過程で開始ノードへの経路を記録していく．

(2) 宛先ノードがRREQメッセージを受信すると経路応答(以下RREP)メッセージを送信する．RREPメッセージはRREQメッセージが辿ってきた経路を逆向きに開始ノードへとユニキャストで中継される．その過程で各ノードは宛先ノードへの経路を記録する．RREQとRREPメッセージは同じ経路を通るので最終的に利用されるのは双方向のリンクだけである．

リアクティブ型のアドホックルーティングの安定性と性能を議論する上で問題となるのは（1）である．DYMOでは全てのメッセージにバージョン番号を含めているので古いRREQメッセージは中継せずに破棄する事が出来るが，それでもRREQメッセージの受信ノードは1度以上それを再送する単純なフラッディングである．

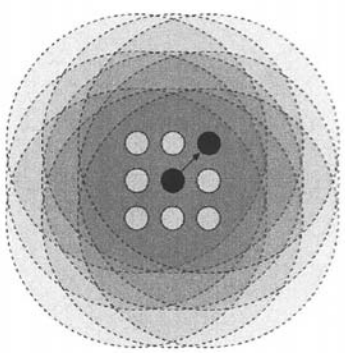

図1 密なネットワークにおけるフラッディング

最も極端な例として図1を考える．全てのノードがお互いの通信可能範囲に入っている密なネットワークである．中央のノードSが右上のノードTへ向かってRREQメッセージを送信する場合，ノードSがRREQメッセージが送信するとノードTだけでなく全てのノードが受信する．各ノードは他ノードの受信状況を知るすべがないため，RREQメッセージを再送してしまい，さらに最大8つのメッセージ複製を受信することになる．これらの冗長なメッセージは全く利用されず単純に破棄されるだけである．また，ソースルーティングのDSR以外のリアクティブ型アドホックルーティングプロトコルでは1回のメッセージ交換で確立される経路は1組だけであり，ネットワーク内で複数の通信が開始される場合にはその通信毎に別のフラッディング処理を行なわなければならない．

以上のように単純なフラッディング手法を利用するリアクティブ型のアドホックルーティングでは大量のルーティングメッセージが発生する危険性がある．この影響はMAC層の競合オーバーヘッドや，電波干渉によってメッセージが壊れて結果としてフラッディング処理が再試行されることにより増幅され，ネットワーク全体の挙動を不安定なものにしてしまう．

一方，OLSR[5]などのプロアクティブ型のルーティングプロトコルでは1つのメッセージで複数の経路をまとめて扱っており，また定期的なメッセージ交換により得られたトポロジー情報を生かして効率的なフラッディングを行なっており，このような問題は起きにくい．

3. 提案手法

これらの問題点を受けて本研究ではリアクティブ型アドホックルーティングの，無通信時のオーバーヘッドがないという長所を生かしたままルーティングメッセージのフラッディング処理を効率化する手法を提案する．提案手法は以下の2つの処理からなる．

(1) 定期メッセージを利用しない近隣探索
(2) 冗長なメッセージ再送の抑制

(1) では各ノードが DYMO のメッセージに1ホップ近隣ノードの情報を付加することで，定期メッセージを使わずに1ホップ近隣ノードと2ホップ対称近隣ノードの情報を収集する．(2) では (1) で得られた情報を元に，各ノードは自分が再送しなくても全ての1ホップ近隣ノードに RREQ メッセージの配送が期待出来る場合には再送処理を抑制する．提案手法をリアクティブ型のアドホックルーティングプロトコルである DYMO に拡張として設計・実装したものが DYMO-ND である．以下では DYMO-ND の拡張部分の処理について詳細に説明する．

3.1 定期メッセージを利用しない近隣探索

3.1.1 近隣ノードデータベース

DYMO-ND では各ノードは新たに近隣ノードデータベースの管理を行なう．近隣ノードデータベースは1ホップ近隣ノード表と2ホップ対称近隣ノード表の2つからなる．

- 1ホップ近隣ノード表
  $\{n\_neighbor, n\_heard\_timeout, n\_sym\_timeout\}$
  n_neighbor は1ホップ近隣ノードのアドレス，n_heard_timeout は n_neighbor から自分に少なくとも片方向リンクが存在すると見なす時間，n_sym_timeout はこの n_neighbor との間に双方向リンクが存在すると見なす時間，をそれぞれ表している．
- 2ホップ対称近隣ノード表
  $\{n2\_neighbor, n2\_twohop, n2\_timeout\}$
  n2_neighbor はこの情報を送信した1ホップ対称近隣ノードのアドレス，n2_twohop は n2_neighbor を経由して到達出来る2ホップ対称近隣ノードのアドレス，n2_timeout はこのエントリを無効なものとして消すまでの時間，をそれぞれ表している．

n_sym_timeout が現在の時間よりも後ならばそのエントリは双方向リンク，n_heard_timeout が現在の時間より後ならばそのエントリは片方向リンクと考える．どちらの時間より後のエントリは無効である．2ホップ対称近隣ノード表には2ホップ対称近隣ノード，すなわち2ホップの双方向のリンクでアクセス出来るノードのみを含んでいる．そのため，時刻が1ホップ近隣ノードのエントリの n_sym_timeout になるとその n_neighbor を n2_neighbor に持つ2ホップ対称近隣ノードのエントリは削除される．

3.1.2 メッセージの拡張

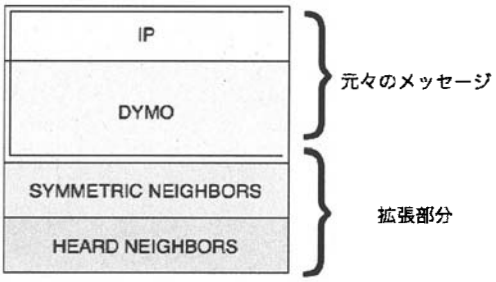

図 2 メッセージフォーマットの拡張

DYMO の各メッセージを図2のように拡張する．DYMO ではメッセージ形式に同じく IETF で提案されている PacketBB[3] を用いている．PacketBB は複数のアドレスを効率的にパックするアドレスブロック形式や拡張性に優れた TLV 形式を採用している．DYMO-ND では元々の DYMO が処理する部分は変更せず，新たに1ホップ近隣ノードのアドレスを集めたアドレスブロックを付加する．近隣ノードアドレスブロックに含まれるアドレスは TLV によって片方向のリンクを持つノードか双方向のリンクを持つノードか区別される．

メッセージの生成

元々の DYMO が生成したメッセージに，近隣ノードデータベースにもとづき近隣ノードアドレスブロックを付加する．

メッセージの処理

元々の DYMO の前にまず近隣ノードアドレスブロックが処理される．したがって DYMO によってメッセージが処理されず破棄される場合でも近隣ノードデータベースは更新される．ノード src から近隣ノードアドレスブロックを受信した時の近隣ノードデータベースの更新手順は以下のように行なわれる．

(1) メッセージを受け取ったことでノード src から自分への片方向リンクがあることが確認出来る．1ホップ近隣ノード表に $n\_neighbor = src$ のエントリを検索し，ない場合は作成する．そのエントリについて $n\_heard\_timeout$ = 現在の時刻 + リンクの寿命時間，と更新を行なう．
(2) 近隣ノードアドレスブロックの中に自分のアドレスが含まれていた場合はノード src との間に双方向リンクがあることが確認出来る．よって $n\_neighbor$ = 送信元アドレスのエントリについて $n\_sym\_timeout$ = 現在の時刻 + リンクの寿命，と更新を行なう．

(3) ここまでの処理で送信元ノードとの間に双方向リンクがあることが確認出来た場合には，近隣ノードアドレスブロックの中の1ホップ対称近隣ノードアドレスそれぞれについて，*n2_neighbor* = 送信元アドレス *n2_twohop* = そのアドレス，であるエントリを検索しなければ作成する．そのエントリに対し *n2_sym_timeout* = 現在の時間 + リンクの寿命，と更新を行なう．

3.1.3 例

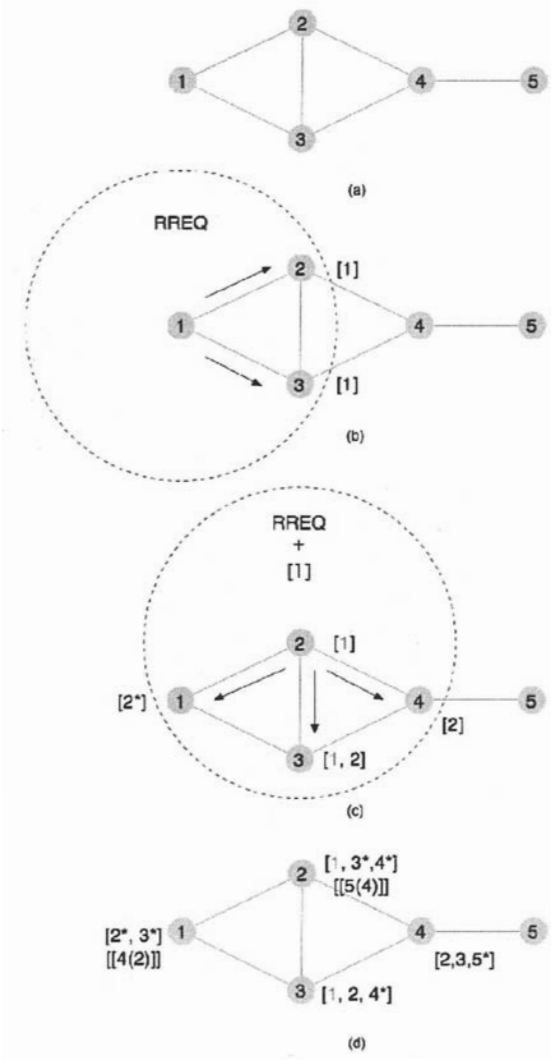

図 3 近隣探索の例

例を使って近隣ノード探索の処理を説明する．図3(a) はネットワークの初期状態を表している．ノード間の実線は双方向リンクを表しており，各ノードの経路表と1ホップ近隣ノード表と2ホップ近隣ノード表は空に初期化されている．ノード1から5へノード1，2，3，4の順にRREQメッセージがフラッディングされ，ノード5から1へノード5，4，2とRREQメッセージが返されたとする．

図3(b) はノード1がRREQメッセージを送信した後の状態である．ノード1の1ホップ近隣ノード表は空なので近隣ノードブロックが空のRREQメッセージが送られている．ノード2，3はノード1からの片方向リンクがある事を知り，1ホップ近隣ノード表にノード1のエントリを加える．図の1重の大カッコに囲まれた数字は1ホップ近隣ノード表を要約したものである．素の数字は片方向リンクを，*が付いた数字は双方向リンクを表している．図3(c) はその後にノード2がRREQメッセージを中継した後の状態である．近隣ノードブロックは1から2への片方向リンクがある事を知らせている．前のメッセージと同様に，ノード3，4は2からの片方向リンクがある事を知り，1ホップ近隣ノード表にエントリを加える．ノード1は近隣ノードアドレスブロックに自分が含まれているので1ホップ近隣ノード表のノード2のエントリを双方向に変える．メッセージが中継される場合には受信したメッセージの近隣ノードアドレスブロックを処理し，更新された近隣ノードデータベースを元にメッセージの生成は生成される．つまり，基本的には内容が変わらずに中継されるDYMOのメッセージと異なり，近隣ノードアドレスブロックは1ホップ毎に中継のデータベースを元にしたものに置き換えられる．このように近隣情報を収集していき，RREQとRREPメッセージの交換を終えた後の1ホップ近隣ノード表と2ホップ近隣ノード表の状態は図3(d) のようになる．2重の大カッコに囲まれた数字は2ホップ近隣ノード表のエントリで括弧の外が2ホップ対称近隣ノードで中が1ホップ近隣対称ノードである．このように経路解決の過程で近隣ノードの情報が集まっていくことを確認出来た．複数の経路解決の中で収集を続ければさらに正確な情報が収集出来る．

3.2 冗長なメッセージ再送の抑制

3.2.1 近隣ノードの推定

DYMO-NDでは全ての1ホップ近隣ノードが実際にメッセージを受信した事や受信出来る見込みが高い事が確認出来た場合にはメッセージの再送を抑制する[4]．配送確認のためにはまず近隣ノードの全体集合を推定する必要がある．NHDP[4] とは異なり，定期メッセージがない近隣ノード探索では定常的に有効な近隣ノード情報を持っている訳ではないため，合わせて近隣ノード情報の収集状態を合わせて管理する．図4が近隣ノード情報の収集状態の遷移図である．3つの状態の説明は以下の通りである．

UPDATE-NEEDED 一定時間以上，ルーティン

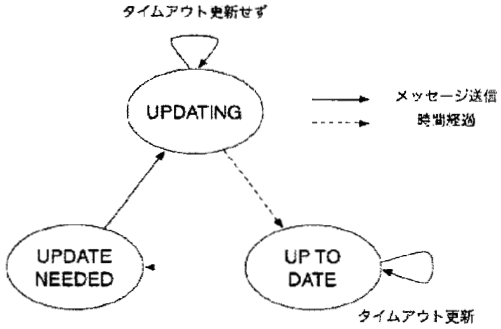

図 4 近隣ノード集合の収集状態

グメッセージを送信しておらず近隣ノード上で自分が近隣ノードデータベースから消えている，もしくは消えかかっている状態．

UPDATING メッセージを送信し返信メッセージを待っている状態．

UP-TO-DATE 返信メッセージを一通り受信し最新の状態を持っている状態．

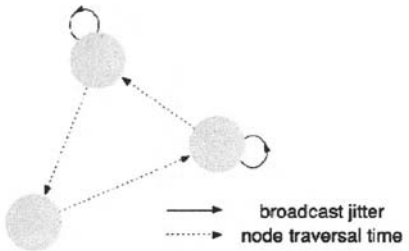

図 5 近隣ノードの情報の収集にかかる時間

ノードの収集状態は初め UPDATE-NEEDED 状態である．メッセージを送信すると UPDATING 状態に変化する．UPDATING 状態に変化した後一定時間は待って UP-TO-DATE 状態に遷移する．UPDATING 状態に留まる時間は 2 ホップを経て情報が集るまでの時間であり，図 5 より最大で $NodeTraversalTime \times 3 + BroadcastJitter \times 2$ と見積もる事が出来る．UP-TO-DATA 状態中に一定時間以上メッセージが送信されなかった場合は UPDATE-NEEDED 状態に戻る．また，UP-TO-DATE 状態に留まる時間は DYMO が使わない経路を無効にするまでの時間より短くした．

3.2.2 メッセージ再送抑制アルゴリズム

冗長なメッセージ再送の抑制は以下のアルゴリズムで実現される．アルゴリズムの入力は，そのノードのアドレスと収集状態，送信元ノードのアドレスとメッセージに含まれる近隣ノードブロックである．出力は再送フラグである．ノードはルーティング処理によって再送が決定され，かつ再送フラグが立っている場合のみメッセージを再送する．

(1) 状態が UPDATE-NEEDED か UPDATING だった場合は再送フラグを立てて判定を終了する．

(2) 送信元ノードが 1 ホップ対称近隣ノードでなかった場合は再送フラグを立てて判定を終了する．

(3) 現在の 1 ホップ近隣ノード表に含まれる近隣ノードの集合 N に対して

(a) 送信元ノード src を N から取り除く
$$N = N - \{src\}$$

(b) メッセージに含まれる 1 ホップ対称近隣ノードブロックを N から取り除く．
すなわち，$N = N - \{addr \mid addr \in msg.sym\_neighbor\_list\}$

(c) メッセージに含まれる 1 ホップ対称近隣ノードブロックのエントリの中で自分のアドレスより小さいもの addr について，近隣ノードデータベースの 2 ホップ対称近隣ノード表から $n2\_neighbor = addr$ であるエントリを探し，$n2\_twohop$ を N から取り除く．
すなわち，$N = N - \{e.n2\_twohop \mid e.n2\_neighbor \in msg.sym\_neighbor\_list, e.n2\_neighbor < my\_addr\}$

以上の処理を経て 1 ホップ近隣ノード集合 N が空集合になっていなければ再送フラグを立てる．

ステップ（1）では，近隣ノードの情報が不完全な状態でアルゴリズムが適用されてしまうことを防いでいる．ステップ（2）で，メッセージの送信元ノードとの間に双方向リンクを持っていないと判定された場合は，送信元ノードがしばらくの間このノードの近隣ノード情報を受け取っていない場合である．メッセージを送信することで送信元ノードの近隣ノードデータベースを更新する．ステップ（3）ではこのノードの 1 ホップ近隣ノードについてそれぞれメッセージの配送経路があるかどうかを判定していく．

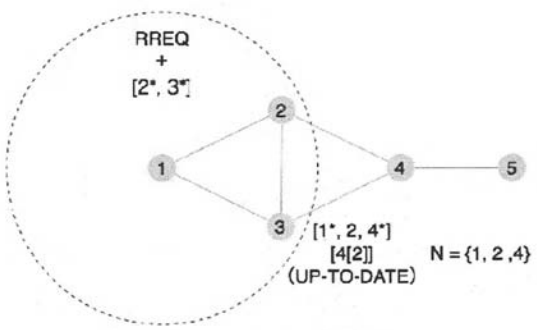

図 6 冗長なメッセージ再送抑制の例

広さ	$1500m \times 1500m$
MAC プロトコル	802.11b
通信レート	11Mbps
通信可能距離	283.6m

表 1　実験環境

Net Diameter	10
Node Traversal Time	40ms
Broadcast Jitter	10ms
Valid Route Time	5s
Delete Route Time	25s
RREQ Retries	3

表 2　DYMO, DYMO-ND の共通パラメータ

3.2.3　例

図 6 を例として再送抑制アルゴリズムを説明する．ノード 3 は収集状態が UP-TO-DATE であり，ノード 1 は 1 ホップ対称近隣ノードなのでアルゴリズムのステップ（1）（2）を通過する．ステップ（3）でまずノード 3 は近隣ノード集合 N_3 を 1, 2, 4 に初期化する．ステップ（3 a）で RREQ メッセージの送信ノードであるノード 1 が N_3 から取り除かれる．

$$\{1,2,4\} - \{1\} = \{2,4\}$$

次にステップ（3 b）では，メッセージに含まれた 1 ホップ対称近隣ノードである 2, 3 が N_3 から取り除かれる．

$$\{2,4\} - \{2,3\} = \{4\}$$

最後にステップ（3 c）で，メッセージに含まれた自分よりアドレスが小さいメッセージに含まれる 1 ホップ対称近隣ノード 2 からアクセス出来る 2 ホップ対称近隣ノード 4 が取り除かれる．

$$\{4\} - \{4\} = \{\}$$

近隣ノード集合 N_3 が空になるのでアルゴリズムは再送フラグを立てずに終了する．したがってノード 3 は RREQ メッセージを再送しない．ステップ（3 c）のアドレスの順序付けによってノード 2 の近隣ノード集合 N_2 からはノード 3 の 2 ホップ対称近隣ノードが取り除かれないので，ノード 2 と 3 が同時に再送を抑制してしまうことはない．

4.　実装・評価

DYMO-ND をネットワークシミュレータ Qualnet[1] 上に実装した．実験環境は表 1 に，に DYMO-ND と DYMO の共通パラメータは表 2 に示す．

DYMO-ND 独自のパラメータである近隣ノードを無効化するまでの時間，UP-TO-DATE 状態から UP-DATE- NEEDED 状態になるまでの時間は Valid Route Time よりも短い 3 秒に，UPDATING 状態から UP-TO-DATE 状態になるまでの時間は少々節 3.2.1 の説明により Node Traversal Time と Broadcast Jitter から計算し 160 ミリ秒に設定した．

Qualnet の DYMO 実装はデフォルトでリング探索が有効になっているが Internet Draft 内ではオプションとして定義されていることと本手法の効果を切り離したかったことから無効にした．また，近隣ノードの情報を経路表にも入れることも可能でこの場合は 2 ホップ内のノードとの通信に RREQ メッセージのフラッディングをする必要がなくなるのでルーティングオーバーヘッドを大幅に削減出来る．しかし，本研究では提案したフラッディングの効率化の部分の評価に注目するためにやはり無効にした．

まず，短時間に多数の経路確立が試みられた場合の DYMO と DYMO-ND の両プロトコルの安定性を比較するために，ノード数が 60 と 100 台の時それぞれについて通信開始間隔を変えながら DYMO に対する DYMO-ND の発生した RREQ メッセージ数比と確立に成功した経路数比，確立された経路あたりのメッセージ数（以下コスト）比を図示した結果が図 7 と図 8 である．

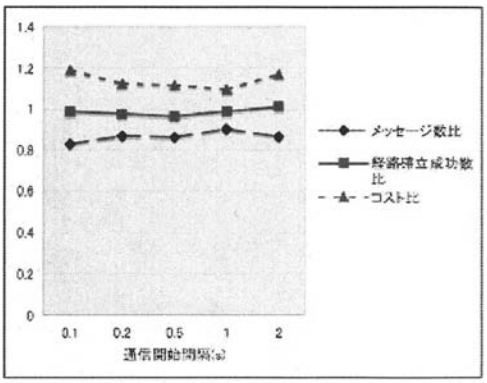

図 7　RREQ メッセージ数・経路確立成功数・コストの比較 (60 台)

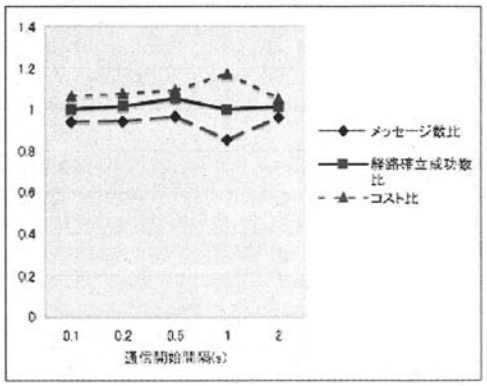

図 8　RREQ メッセージ数・経路確立成功数・コストの比較 (100 台)

フラッディングされたRREQメッセージ数は全ての通信開始間隔においてDYMO-NDがDYMOより少なく最大で17%のRREQメッセージが削減されている．60台の場合の方が削減率は高かった．理論的にはxノード配置が密な場合の方がメッセージ削減率は多いはずだがアルゴリズムのステップ（3）において100台の場合には近隣ノード集合Nに残してしまうノード数が平均的に多く，特にノード配置が密な場合に向けてアルゴリズムを改良する余地が残っていることが考えられる．一方，通信の確立成功率についてはDYMO-NDがDYMOより100台の場合も60台もDYMOとほとんど変わらないという結果になった．最後に確立成功率はほとんど変わらないままメッセージ数を減らすことが出来ているので，コスト比は総じてDYMO-NDが優れているという結果になった．RREQメッセージが効果的に配送されていることが確認出来た．

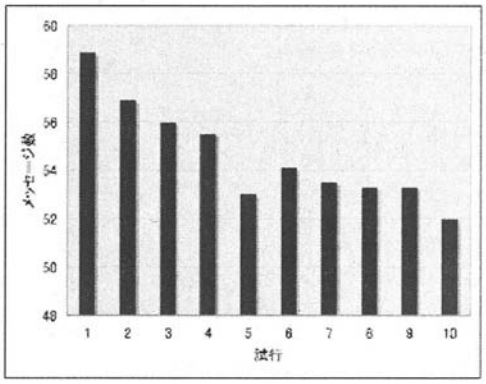

図9 フラッディングの効率化 (60台)

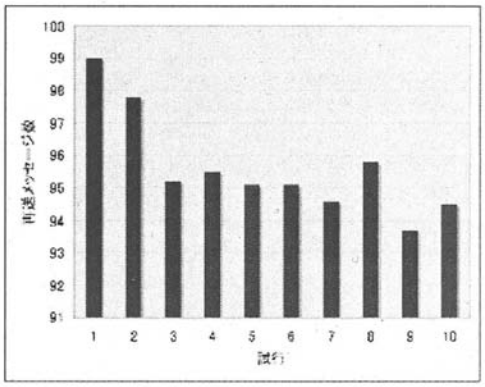

図10 フラッディングの効率化 (100台)

図9と図10はノードをランダム配置しある1台のノードからネットワーク内に存在しないノードへRREQメッセージを10回送った時の各RREQメッセージが再送された回数である．各RREQメッセージの送信開始間隔は0.5秒である．またRREQフラッディングの再試行は無効にしている．どちらも1回目のRREQメッセージは開始ノード以外の全てのノードがメッセージを再送しているが，2回目以降は再送抑制アルゴリズムが働き再送メッセージ数が減少している．特に60台の場合は3回目以降，100台の場合は5回目以降ではメッセージ数を安定し始めている．ノードが密な程近隣ノードの収集は早く抑制の効果も早く観測されていることが分かる．また6回目以降は最初のRREQフラッディングで得られた近隣ノード情報の有効期限が切れ始めているが，その後も安定してフラッディングを抑制し続けるが確認出来た．

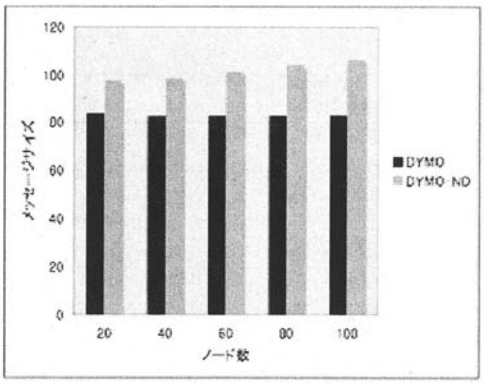

図11 平均メッセージサイズ

最後に手法のオーバーヘッドを測定した．DYMO-NDでは近隣ノードアドレスブロックを含む分，DYMOよりもメッセージサイズが増加している．図11はノードの配置密度を変えながら平均メッセージサイズを測定したものである．ノードの配置が密になればなるほど近隣ノードアドレスブロックに含まれるノード数が増加するため，メッセージサイズは大きくなる傾向にあり，ノード数が100台の場合には22%程度増加している．メッセージ数が減少すると同時にメッセージ数が増えてしまってはいるが，MAC層の競合オーバヘッドを考えるとメッセージサイズよりもメッセージ数の影響が大きいので，本手法はリアクティブ型ルーティングプロトコルの不安定性を解消する上で有効だと考えられる．

5. 関連研究

Pleischらは既存の効率的なメッセージフラディン

グ手法について，定期的なメッセージ交換により取得した近隣のノード情報をもとにフォワーディングするノードを選択するオーバーレイ型アプローチと，近隣のノード情報を用いずに既に受信したメッセージの数や何ホップ先へのメッセージであるかをもとに自立的にフォワーディングする確率を計算するローカル情報型アプローチとに分類している[11]．

前者の代表的な例として，OLSR[5]，ZRP[6]，CBRP[7]などが挙げられる．これらのルーティングプロトコルでは隣接ノード間で定期的に HELLO メッセージを交換し，それに近隣ノード情報を収集する．代表的なフラッディング抑制手法として，OLSR では 各ノードが自身の全ての２ホップ対称近隣ノードに到達するために必要な１ホップ対称近隣ノードのみにフォワードを許可しておりこのノードを MPR と呼ぶ．少数の MPR 集合のみが中継処理を行なうので効率的にメッセージをフラッディングすることが出来る．オーバーレイ型アプローチの問題点として，ノードの参加や脱退により近隣ノードの情報を再取得し MPR 集合を再構築する間，通信の信頼性が低下すること，またノードの通信の有無によらず定期的な HELLO メッセージの交換が必要であり通信資源を消費することが挙げられる．一方，ローカル情報型アプローチでは定期的なメッセージの交換は不要であり，ノード内の情報だけで中継するかどうかを決定する手法がとられている．Ni らはではメッセージを再送のするかどうかの判定に，受信したメッセージ数を数えるカウンタ型や，確率的で中継するかどうかを決める確率型，メッセージの送信元ノードとの距離をもとに決める距離型などが提案している[9]．瀬山らははあらかじめノードをグループ分けしておき異なるグループのメッセージの中継を抑制している[12]．ローカル情報型アプローチの問題は適切なパラメータ設定が静的に行なわれていることである．適切なメッセージ数や距離のしきい値や確率の値，グループの分け方を決定するためには事前に適用されるネットワークの情報が必要になる．

6. おわりに

本研究ではリアクティブ型アドホックルーティングの欠点であるルーティングメッセージの非効率なフラッディングを解決するため定期メッセージを用いない近隣ノード探索とそれを利用した冗長なメッセージ再送を抑制するアルゴリズムを提案し，シミュレータ上で実験・評価を行なった．提案手法は経路確立成功率を保ちながらネットワークを流れるルーティングメッセージの総数を最大で 17%削減することを確認した．一方メッセージサイズは増加してしまうが，MAC 層のオーバーヘッドを考えるとメッセージ数の方がメッセージサイズより影響が大きいと考えられる．また，近隣ノードの情報を経路としても利用する手法など今回は無効にした機能などを考慮するとまだ改善の余地は残っている．今後はプロトコルの詳細なトレースを取りアルゴリズムの検証と改良を行ないたい．

参考文献

1) Qualnet Network Simulator 4.0. `http://www.qualnet.com/`.
2) I.Chakeras and C.Perkins. Dynamic MANET On-demand (DYMO) Routing IETF Request For Comments(RFC).
3) T.Clausen, C.Dearlove, J.Dean, and C.Adjih. Generalized MANET Packet/Message Format, draft-ietf-manet-packetbb-04 IETF Internet Draft(ID).
4) T.Clausen, C.Dearlove, J.Dean, and The OLSRv2Design Team. MANET Neighbor Discovery Protocol, draft-ietf-manet-nhdp-04 IETF Internet Draft(ID).
5) T. Clausen, P. Jacquet, and Project Hipercom. Optimized Link State Routing Protocol (OLSR) IETF Request For Comments(RFC).
6) ZygmuntJ. Haas, Marc R. Pearlman, and Prince Samar. The Zone Routing Protocol (ZRP) for Ad Hoc Networks, draft-ietf-manet-zone-zrp-04.txt IETF Internet Draft(ID).
7) Mingliang Jiang, Jinyong Li, and Y.C. Tay. Cluster Based Routing Protocol(CBRP), draft-ietf-manet-cbrp-spec-01 IETF Internet Draft(ID).
8) D. Johnson, Y. Hu, and D. Maltz. The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4 IETF Request For Comment(RFC).
9) Sze-Yao Ni, Yu-Chee Tseng, Yuh-Shyan Chen, , and Jang-Ping Sheu. The Broadcast Storm Problem in a Mobile Ad Hoc Network. *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, 1999.
10) C. Perkins, E. Belding-Royer, and S. Das. Ad hoc On-Demand Distance Vector (AODV) IETF Request For Comments(RFC).
11) Stefan Pleisch, Mahesh Balakrishnan, Ken Birman, and Robbert van Renesse. Mistral: Efficient flooding in mobile ad-hoc networks.
12) Toshiya Seyama and Hiroaki Higaki. G-AODV : Group-Based Extension of AODV Routing Protocol. In *IPSJ SIG Notes*, 2006.