モバイルコンピューティングとワイヤレス通信　20－32
(2002.　3.　8)

アドホックネットワークに適した名前解決機構 RANR（ランダム割り当て名前解決機構）

青木 基衣† 斉藤 匡人† 間 博人§ 徳田 英幸‡§
†慶應義塾大学 総合政策学部 ‡慶應義塾大学 環境情報学部
§慶應義塾大学大学院 政策・メディア研究科
〒 252-8520 神奈川県藤沢市遠藤 5322
E-Mail: {motoi,masato,haru,hxt}@ht.sfc.keio.ac.jp

近年，既存のインフラストラクチャに依存せずに，マルチホップ通信機能を持つ無線端末同士が即興的に構築するモバイルアドホックネットワークの研究に注目が集まっている．しかし，アドホックネットワーク構築後に必要とされる，有線インターネットにおける DNS (Domain Name System) のような，名前解決サービスに関する研究は行われていない．そこで本稿では，アプリケーション構築を支援するための基盤技術となる名前解決機構 RANR を提案する．RANR は，ネットワーク内でユーザの好みに基づいた一意なホスト識別子となる名前割り当てを行い，その名前を用いた通信機構をアプリケーションに提供する．本稿では，アドホックネットワークを構築するためのルーティングプロトコルの一つである DSR (Dynamic Source Routing) を用いて 8 台のノードで構築したアドホックネットワーク上で RANR を実装し評価を行った．評価結果から，RANR は名前解決に起因するオーバヘッドを最小限にとどめながら，現実的な時間内で名前解決を実現しつつ端末数の増加に対する十分なスケーラビリティを持つことが示された．
キーワード：モバイルアドホックネットワーク，名前解決，DNS

Name Resolution System for Mobile Ad Hoc Networks RANR (Randomly-Assigned Name Resolution)

Motoi Aoki† Masato Saito† Hiroto Aida§ Hideyuki Tokuda‡§
†Faculty of Policy Management
‡Faculty of Environmental Information
§Graduate School of Media and Governance
5322 Endo, Fujisawa, Kanagawa 252-8520, Japan
E-Mail: {motoi,masato,haru,hxt}@ht.sfc.keio.ac.jp

Due to recent advancement in technology of wireless communication, use of small sized computers in a mobile environment has become possible. Mobile ad-hoc networks are temporarily constructed by mobile hosts without established infrastructure. Such networks can constitute a base for human cooperative works as well as for communications between information systems. However, there have not been sufficient work of name resolution systems suitable for ad-hoc networks. In this paper, we propose a name resolution system called RANR suitable for ad-hoc networks as a base technology for application construction environment. RANR can provide a node with a unique user-oriented names in an ad-hoc network, thereby adapting to dynamic changes of network identifier configuration. In addition, RANR relieves users and their applications from the necessity of identifying the network addresses (IP addresses) of nodes by introducing unique names as employed in DNS. We have designed and implemented RANR in FreeBSD, and evaluated the performance in an ad-hoc network system built with PCs running Dynamic Source Routing (DSR) protocol.
KeyWords : Mobile Ad Hoc Networks, Name Resolusion System, DNS

1. はじめに

近年，無線通信機能を持つ情報端末がルータの役割を担いパケットを中継することで，即時的にネットワークを構成するモバイルアドホックネットワーク (MANET) に関する研究が盛んに行われている．MANET では，中間端末のマルチホップ機能により，直接通信できない端末同士でもネットワークを組み，パケットを送受信することが可能となる．そのため，既存のインフラストラクチャを必要とせず，無線通信を用いて様々な場所でネットワークを構築することができる．この特徴から，災害地での利用，また会議などに持ち寄った端末同士で組むネットワーク，タクシー，バス等で構築される交通ネットワークなどの利用方法が考えられる．MANET を構築するための経路制御プロトコルも多く提案されており，MONARCH プロジェクト [1] の DSR (Dynamic Source Routing)[2] や，AODV (Ad Hoc On-Demand Distance-Vector Routing)[3]，TORA (Temporally Ordered Routing Algorithm)[4] などが挙げられる．

こうしたネットワークは，情報機器同士のみならず人間同士の協調作業を行うためのアプリケーション基盤になると予想される．そのため，インターネットで利用されているように情報端末に識別子としての名前を付け，ユーザに対してより使いやすいネットワーク環境を構築することは重要である．現在インターネットでは DNS (Domain Name System)[5] と呼ばれる名前解決機構が広く用いられている．このシステムにより IP アドレスに分かりやすく覚えやすい文字列型のホスト名が対応付けられている．そのためユーザは IP アドレスを意識することなくホスト名だけでの通信が可能であり，Web などのアプリケーションでも用いられている．

しかし，MANET は既存の有線ネットワークとは異なり，ネットワークリンク・トポロジが不安定であり IP アドレスがノードに必ずしも固定されているとは限らないため，インターネットで用いられている名前解決機構をそのまま利用するだけでは不十分である．MANET で名前解決を行うためには，この特徴に応じた名前解決機構が必要になるが，現在はそのような機構についてあまり研究されていない．本稿では，アプリケーション構築を支援するための基盤技術としてアドホックネットワークに適した名前解決機構を提案する．

本稿の構成として，まず MANET における名前解決の問題点を分析し，次にそれを解決する幾つかの手法とそのトレードオフについて述べる．そして「名前のランダム割り当て」手法を用いた RANR の設計と実装について述べ，最後に本システムのスケーラビリティを考察し，オーバヘッドと名前解決に要する時間を評価する．

2. MANET における名前解決

MANET を利用する場合，まず通信しようとする相手のノードを特定する必要があるが，ネットワークを組むたびにユーザが通信相手の IP アドレスを確認して互いを認識する方法は現実的ではない．なぜなら，MANET ではあるノードが常に同じ IP アドレスを使用できるとは限らないため，一つの IP アドレスで必ずしも同じノードを特定できないからである．MANET における IP アドレスは，IP 通信を行うために一時的に各ノードに一意に割り当てられるため，ノードに IP アドレスを固定することはできない．さらにネットワークの規模が大きくなるとすべての IP アドレスを確認して相手を特定することは非現実的である．

そこでユーザの利便性を向上させるために，通信相手を特定するための容易に認識できる識別子をノードに割り当てる必要が生じる．これを実現するのが MANET における名前解決機構である．本章ではまず名前解決機構を定義し，次に MANET で名前解決をする際の問題点を示す．最後に既存のシステムで対応する場合の限界について述べる．

2.1 名前解決機構の概要

名前解決機構とは，人間にとって識別が容易な情報端末識別子を用意し，それをネットワークプロトコルのための識別子に対応付けするシステムのことである．例えば DNS では，文字列型のホスト名と呼ばれる識別子を用意し IP アドレスに対応付けている．このシステムにより通常ユーザはホスト名だけを意識して通信を行うことが可能になる．本稿でも以後 IP ネットワークを想定して，IP アドレスに名前を対応付けるための名前解決機構について述べる．

2.2 MANET における名前解決の問題点

MANET は，無線を通信媒体として動的に構成されるネットワークであるため，名前解決の際に次のような問題が生じる．

(1) 通信リンクの品質やネットワークトポロジの動的な変化，ノードの出入りが頻繁に起こるなどの理由により，名前解決を行う中央管理サーバを想定することができない．

(2) IP アドレスがノードに固定されていないため，IP アドレスと名前の対応情報を静的に保持することができない．そのため名前を何らかの方法で動的に割り当てる必要がある．

(3) 名前を動的に割り当てた場合，ネットワーク内で名前の衝突が起きる可能性がある．

まず，MANET では通信品質とネットワークトポロジの不安定さのため中央管理システムを設置することが困難である．MANET ではノードの移動に伴い，ネットワークリンクの状態が常に変化しているのでサーバとなったノードと常にリンクが保たれていることは保証されない．またトポロジも頻繁に変化するため，サーバとなったノードが常にネットワーク内に存在するとは限らない．これらの理由から完全に中央管理機構に依存する設計では不十分である．

第二の問題として IP アドレスと名前の対応情報を静的に保持できないことが挙げられる．これは同じノードが常に同一の IP アドレスを使用できるとは限らないためである．同じ理由で，名前を静的に割り当てることにも問題があるので，名前解決機構が分散的，動的な方法で名前を割り当てる必要がある．

しかし，これが三番目の問題につながる．すなわち，ネットワーク内で動的に名前を割り当てると名前の衝突を引

き起こしてしまう可能性がある．ある時点では一意な名前であっても，新しいノードが追加された時や，他のアドホックネットワークと合併した時に名前の衝突が生じる可能性がある．そのため，名前解決機構は名前の衝突に動的に対応できなければならない．

2.3 既存の名前解決システム

既存の名前解決機構として広く用いられているシステムにDNSがある．これは現在のインターネットで広く利用されている．DNSは，静的なネットワーク構成を想定しているため，階層化された中央管理サーバがIPアドレスと名前の対応付け管理テーブルを静的に保持している．しかし，MANETでは中央サーバに静的な設定ファイルを置くという手法では不十分である．

DNSを動的なネットワーク環境に対応させたものとしては，DDNS (Dynamic Domain Name System)[6]が挙げられる..これは，DNSサーバ内のテーブルを自動的に更新するシステムで，1998年初頭ごろから実装が進み，現在実用段階に入っている．DDNSは，元々DHCP (Dynamic Host Configuration Protocol)[7]やダイアルアップ接続サービスを利用している場合などの，ネットワークに接続するたびにIPアドレスが変わってしまう環境で，端末に固定の名前を割り当てることを目的としていた．これは，端末がIPアドレスを振られるたびにそれをDDNSサーバのデータベースに自動的に登録することで，従来のDNSが提供しているのと同じ機能を提供するシステムである．

DDNSが想定している，IPアドレスが毎回一定でないという特徴はアドホックネットワークの特徴と類似している．そこで，このDDNSのサーバをネットワークのすべてのノードに実装することでMANETにおける名前解決を行うことが考えられる．しかし，この方法を実現するには幾つかの課題が残る．まず，ネットワーク内のDDNSに登録するため，それらサーバを見つけるための手法が必要になる．また，ネットワークリンク・トポロジが不安定なネットワーク内のサーバ同士で名前の一貫性を保つ必要もある．

さらに，ネットワーク内に設けるべきサーバの数も考慮しなければならない．もしすべてのノードがサーバにならないならば，サーバにするノードを動的に選出する方法が必要になる．またもしすべてのノードがサーバになるなら，各ノードが自分で名前情報を管理することと同じになり他のサーバに聞く必要性は低下する．すなわち，アドホックネットワーク内でDDNSを用いても，MANETにおける名前解決機構の根本問題を解決できるわけではない．

また，MANET内で衝突しない識別子を割り当てるという点ではIETFのzeroconfワーキンググループ[8]が提案しているzeroconfネットワークが類似している．これは既存設定の存在しないネットワークでリンク層のIPv4アドレスを割り当てるものである．zeroconfネットワークも中央管理サーバが存在しない環境を想定している．

3. MANETに適した名前解決機構

MANETに適した名前解決機構を実現するためには，前章で述べた問題に対応する必要がある．本章では，これらの問題に対応可能な名前解決機構としてRANR（ランダム割り当て名前解決機構）を設計する際に，利用する名前のデータ構造，名前の割り当て方法，名前の管理方法に関してそれぞれ選択できる手法とそのトレードオフについて述べる．

3.1 名前の構造

名前のデータ構造には以下の二種類が考えられる．

- フラットなデータ構造
- 階層化されたデータ構造

まず，単純な文字列データを名前として用いる方法がある．この場合，名前はフラットな構造になる．これは具体的には'dari'などのシンプルな文字列のことである．そしてもう一つは構造化されたデータを名前として用いる方法である．この場合名前は階層化される．具体的には'dari.sfc.keio.ac.jp'というインターネットで用いられているような名前である．

ネットワークの規模が大きい場合は階層化されたデータを名前として用いる方が現実的である．名前が階層化されていれば，名前を管理するデータベースも階層化することが可能になり大規模なネットワークでも管理が容易になる．またMANETで階層化された名前を用いることには，以下の長所が考えられる．

- 名前を論理的に分類することによってネットワーク自体を階層的に扱える
- 物理的な情報を名前に含めることによってネットワークの利便性を高める

MANETはIPプライベートネットワークとして構築されるため，そのままではフラットな構造のネットワークである．しかし，階層化された名前を用いることによって複雑なグルーピングをしなくてもノードを階層的に扱うことが可能になる．例えば，論理的に与えられた位置情報や所属情報等を名前に含めることが考えられる．

また，位置情報や提供しているサービス情報等の物理的な情報を名前に含めることによってネットワークの利用性を高めることが可能になる．例えば端末がGPS情報を取得できる場合，その位置情報を利用することが考えられる．これにより，名前は単に相手を特定するだけでなく付加的な情報も含んだ名前システムの構築を支援するものになるが，同時に名前解決機構の正確さや迅速さを犠牲にする可能性がある．階層的な名前の他の弱点としては，名前解決のためにシステムにかかる負荷が高くなることや，完全な分散型システムとして実装するのが困難であることが挙げられる．また階層化された名前は，ノードの移動が頻繁に起こり，ネットワークトポロジが激しく変化するネットワークには不適切である．これは，位置情報の変化やノードの移動のたびに名前を変えるとネットワークへの負荷もかかる上，名前の一貫性を保つことが困難になるためである．

RANRでは接続端末が100台程度のローカルネットワークを想定しているため，階層化された名前ではなくフラットな名前を利用した．比較的小規模なネットワークではフラットな名前構造を用いた方がシステムへの負荷が少なくなるためである．

3.2 名前の割り当て手法

名前の割り当て手法とは，各ノードにどこでどのように

名前を割り当てるかということであり，以下の二つの方法が考えられる．

- 中央管理システムからの割り当て
- 個々のノードによる自律割り当て

適切な名前の割り当て手法は，ネットワークの規模によっても異なる．比較的小規模なネットワークであれば，個々の端末が自律的に好きな名前を自分に割り当てることができるだろう．しかし大規模なネットワークになると，名前が衝突しないようにだれかが名前を管理してそれを割り当てたり，一定のルールに従って名前を決める必要が生じる．

ネットワーク内の中央管理サーバが名前情報を集中的に維持管理し，割り当てを行う手法の最大のメリットは，名前の衝突が生じないことである．また，サーバがネットワーク内のノード情報を一括管理するため，ネットワーク内のノードはサーバの存在だけを知っていればよいことになる．この方法では，従来の DNS も一部使用可能である．

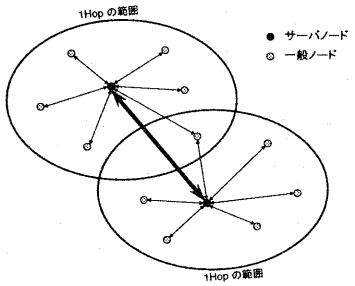

図 1 中央管理システムからの割り当て

MANET では，常にサーバとしたノードとリンクが保たれているとは限らないため，サーバを幾つか分散させて持つ方法も考えられる．図 1 は，MANET 内に名前を割り当てるサーバを幾つか配置して，各ノードは 1 ホップで通信可能なサーバにコンタクトするというモデルを示している．この場合，サーバ同士は互いに既知で名前が衝突しないよう管理されていることを前提とする．

しかしこの手法ではサーバへのリンクが切れた場合の新たなサーバの選出が問題となる．さらにサーバ間の一貫性を保つこと，サーバへのリンクが切れた場合の処理を考慮すると，システム構築のコストに見合うだけの効果を得ることは難しい．中央サーバへ依存した機構は，構成ノードが頻繁に入れ替わるような動的な MANET にはあまり適していない．しかし，特定のノード間で明示的に構築されるような非常に静的な MANET には向いている．

個々のノードが自律的に自分に名前を割り当てる手法は完全な分散型システムとして実現できる．これは，ネットワーク内の各ノードが独自に任意の名前を割り当て，それをネットワークに通知することで実現される．RANR では，名前割り当て手法としてこの方法を採用した．

自律割り当てモデルを簡易に示すと図 2 のようになる．この図では，自分に割り当てた名前を 1 ホップ以内で届く範囲のノードに通知する方法を取っている．しかし自律割り当てを行う場合，名前解決の負荷を最小限に抑えるよう名前情報交換の手法や衝突解決アルゴリズムを設計

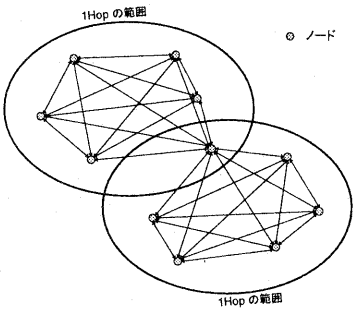

図 2 自律割り当て

する必要がある．これが不適切であるとネットワークに過大な負荷をかけてしまう．また名前が階層化されている場合に，自律割り当て手法を取るとネットワークにかかる負荷が高くなるため，この手法はフラットな構造を持ったシンプルな名前を用いている場合に適している．

自律割り当て手法では中央管理システムが全く存在しないため，ネットワーク内の各ノードは，自分自身の名前情報を他に通知すると共に，他のノードの名前情報をも収集してそのデータを維持管理する必要が生じる．また，最大の問題としてネットワーク内での名前の衝突が挙げられる．RANR では，各ノードにネットワーク内の名前情報を監視する機能を持たせることによりこの問題を解消した．その点については後節で詳述する．

3.3 名前の管理方法

名前の管理方法とは，どこで IP アドレスと名前の対応情報を管理するかということであり以下の二つの方法が考えられる．

- 中央サーバで管理
- 各ノードで管理

管理には，名前と IP アドレスの対応情報の保存，更新作業が含まれる．この保存先となるノードに，名前を知りたい側は問い合わせることになる．これも，小規模なネットワークであれば各ノードがネットワーク上の他のノードの名前情報を管理することができる．しかしネットワークの規模が大きくなるとこの方法で常に最新の情報を管理することは非現実的になる．そのため，ネットワーク内に名前情報のデータベースを持つことが考えられる．DNS では，ホスト名と IP アドレスの対応情報を分散データベースとして管理しており，そのデータベース自体も構造的に管理されている．一般的に，名前の管理方法は名前の割り当て手法がどのようなものかに大きく依存する．RANR では名前割り当て手法として完全分散型の自律割り当て手法を採用したため，名前の管理も個々のノード内で行われる．

4. RANR システムの設計

本章では，RANR が想定する MANET 環境と RANR の概要を述べ，その後 RANR のモジュールについて詳述する．

4.1 RANR が想定する MANET 環境

RANR は動的に構築される MANET の中でも，接続端末数が 100 台程度の比較的小規模なネットワークを想

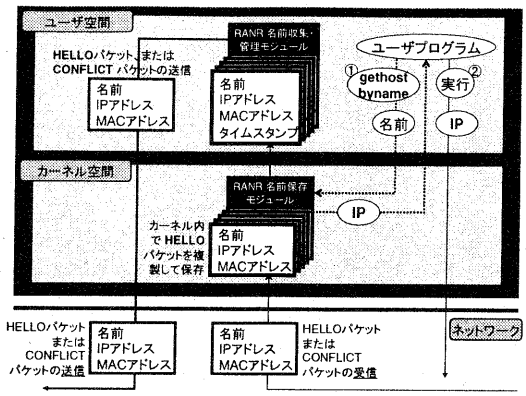

図 3　RANR システムの概観

定している．具体例としては，会議等で持ち寄った端末同士で組むネットワークやバスネットワークなどが考えられる．これらのローカルネットワークでは，ノードとユーザが密接に関連しているため名前解決を行うことが有効であると考えられる．

4.2 RANR の概要

RANR は従来の DNS が実現しているような名前解決サービスをアドホックネットワークにおいて提供する．RANR は次の機能を持つ．

(1)　自ノードへの名前割り当て機能
(2)　他ノードの名前情報管理機能
(3)　ネットワーク内の名前衝突の解決機能
(4)　既存のアプリケーションによる名前利用の提供機能

RANR では，システムに汎用性を持たせるため，これらの機能を二つのモジュールによって実現した．一つは，自ノードへの名前割り当て機能，他ノードの名前情報管理機能，ネットワーク内の名前衝突の解決機能を提供する名前収集・管理モジュールで，もう一つは既存のアプリケーションによる名前利用の提供機能を実現する名前保存モジュールである．前者はユーザ空間に実装され，後者はカーネル空間に実装される．この二つのモジュールの関係を図 3 に示す．

この図は，自ノードへの名前割り当て機能，他ノードの名前情報管理機能，ネットワーク内の名前の衝突の解決機能を実現するため各ノードの名前収集・管理モジュール同士が通信を行って，ネットワーク内の名前を管理していることを示している．また一ノード内では，ネットワーク内の名前情報のコピーを保存している名前保存モジュールが，既存のユーザプログラムに対して名前解決サービスを提供することを示している．次小節ではこれらの各モジュールについて詳述する．

4.3 名前収集・管理モジュール

ユーザ空間に実装された名前収集・管理モジュールは，機能 1 から 3 を二種類のメッセージを交換することで実現する．二つのメッセージとは，ノードが自分の名前を通知するために用いる HELLO メッセージと，名前の衝突が起きたときにそれを知らせるために用いる CONFLICT メッセージである．

自ノードへの名前割り当て

RANR では名前の割り当て手法として「自律割り当て方式」を採用した．ネットワーク内の全てのノードは，独自に任意な名前を付けることができる．自ノードへの名前割り当てを行う際，RANR はフラットな構造のデータを名前として用いる．これは，接続端末数のそれほど多くない小規模なネットワークにおいて最も割り当てコストのかからない名前である．この名前はユーザの好みで指定されるもので，システムが自動的に生成することはない．また名前自体が位置情報等の特定の情報を含むこともない．なお，この名前はネットワーク内で衝突が起きた場合にのみシステムから変更を求められる．

他ノードの名前情報の管理

まず，各ノードはユーザから指定された名前を HELLO メッセージを用いてネットワークに通知し，同時に他のノードから通知される HELLO メッセージの収集も行う．各ノードがこの動作を繰り返すことで，ノードの移動によりトポロジが激しく変動している場合でもネットワーク内の他のノードの名前情報を徐々に集めていくことができる．この動作手順を図 4 に示す．

図 4 と図 5 の状態遷移図で用いられている各状態は以下のように定義される．

待機
パケットの受信を待つ．受信すると true を返す．また n, N 二つのカウンタの管理を行う．
(n) リスト中のエントリのタイムスタンプを走査するための時間カウンタ
(N) HELLO メッセージを送信するための時間カウンタ
(t) リスト中のエントリが古くなっていないか走査するための間隔
(T) HELLO メッセージを送信する間隔

送信
与えられたメッセージタイプのパケットを送信する．

受信
パケットのメッセージタイプによって処理を分ける．

リストの走査
自ノードが保持するリストのエントリを走査する．
- 名前の重複確認
  重複が検知された場合は false を返す．
- MAC アドレスの重複確認
  重複が検知された場合は false を返す．
- エントリの追加
- エントリの更新

自ノードへの名前の割当
ユーザの好みの名前を自ノードに割り当てる．

終了処理
カーネルの処理に戻りユーザ空間にパケットを渡す．

なお，矢印の横には状態が遷移する条件が書かれており Mtype は メッセージタイプを表す．また網掛けされた文字は引数として与えられるパケットの種類を表しており，C は CONFLICT メッセージを，H は HELLO メッセージを示している．

名前通知を行う際のネットワークへの負荷を軽減するため，初期状態の各ノードは HELLO メッセージを送る際にブロードキャストを用いて 1 ホップ内にいるすべての

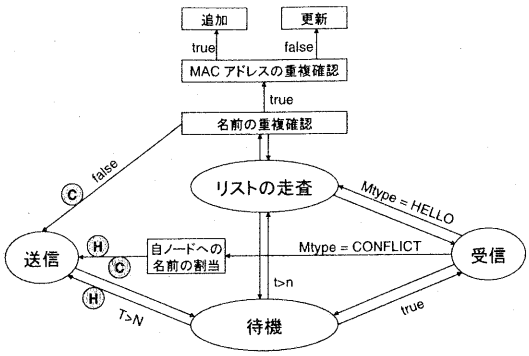

図 4　名前収集・管理モジュール

ノードにパケットを到達させる．そして一定の期間が経過したらブロードキャストを行う間隔を大きくして，代わりに自ノードのリスト内のノードにユニキャストで HELLO メッセージを送信する．これにより，ブロードキャストによるネットワークへの負荷を最小限に押さえ，同時に一度リストに入れたノードは 1 ホップ内から外れても通信を続けることが可能になる．DSR では，ブロードキャストパケットは転送されないが，ユニキャストパケットの場合ルートディスカバリ機能によりマルチホップで届けることが可能だからである．

またこのモジュールでは，リスト中の各エントリにタイムスタンプを付加してそのエントリを管理している．長期間 HELLO メッセージを受信しなくなったノードはネットワークからいなくなったと判断される．

RANR では，名前と現在用いている IP アドレスに加え，MAC アドレスをネットワークに通知する．そして MAC アドレスを指標にノードを識別している．これは，MANET の特性上，IP アドレスも名前もノードへの一意性が保証されていないため，同一ノードを識別するのにノードに固定された別の識別子を必要とするからである．MAC アドレスは衝突がないようハードウェア的に割り当てられたものである．MANET の場合無線を用いているので，無線 LAN インタフェースの MAC アドレスがこれに当たる．しかし MAC アドレスを用いることも，文献 [9] で指摘されているように完全な方法ではない．なぜなら，MANET のノードは必ずしも 48bit の IEEE から割り当てられた一意な MAC アドレスを持ったネットワークインタフェースカードを使用しているとは限らないためである．またインタフェースカードの MAC アドレスをプログラムから書き換えることが可能であることも指摘されている．

ネットワーク内の名前衝突の解決

ネットワーク内で名前の衝突が起きた場合，それを検知した他のノードから CONFLICT メッセージが送信される．衝突した名前を受け取ったノードは，そのエントリのリストへの登録を避けることにより自ノード内での名前の一貫性を保つ．RANR プロトコルでは，CONFLICT メッセージを受信した場合必ず名前を変更しなければならない．図 4 にも示されている通り，CONFLICT メッセージは次の二つの場合に送信される．

- 自ノードと重複した名前を通知する HELLO メッセージを受信した場合
- 自ノードが保持するリスト中のエントリと重複した名前を通知する HELLO メッセージを受信した場合

しかしネットワーク内で名前が衝突した場合，どちらの名前を優先させるかが問題となる．先にその名前を使っていたノードを優先させることが妥当であると考えられるが，MANET においてそれを正確に特定するのは難しい．

そこで RANR では CONFLICT メッセージを受け取る量を調整することでこの問題を解決した．これは，ネットワーク内での認知度を指標にしてどちらが先に名前を用いていたかを決め，先に使用していたノードがその名前を引き続き使用できる確率を高める手法である．具体的には，各ノードが自分の名前と衝突した場合に加えて，自分の保持するリスト内の名前と衝突した場合に CONFLICT メッセージを送る．これにより新しく検知されたノードはそのネットワークでの認知度に応じた量の CONFLICT メッセージの量を受け取ることになる．すなわち，もしあるノードと名前が衝突した場合，そのネットワーク内に自分を既に認識しているノードの数の方が，相手を既に認識しているノードの数より少なければ，自分の方が CONFLICT メッセージを相手より多く受け取ることになり，自分の方が名前を変える確率が高くなる．

しかしこの方法でも確率を調整できるだけで完全に正確に判断できるわけではない．無線の性質上，先に CONFLICT メッセージを受け取った方が名前を変えることになるからである．また，もし名前が解決される前に両方が CONFLICT メッセージを受け取れば，両方が名前を変えることもありうる．

4.4　名前保存モジュール

カーネル空間に実装された名前保存モジュールは，ユーザ空間の他のプログラムに対して DNS のような名前での通信を可能にする．gethostbyname ライブラリ関数に加えられた RANR コマンドが IO コントロール機能を通じてこのリストから該当する IP アドレスを取得しユーザプログラムに渡す．もし該当する名前がない場合 gethostbyname は続く標準コマンドを実行する．

このモジュール自体はネットワークに何かを通知したりリストの管理を行うことはない．カーネルが RANR パケットを受信すると，名前保存モジュールは図 5 に示され

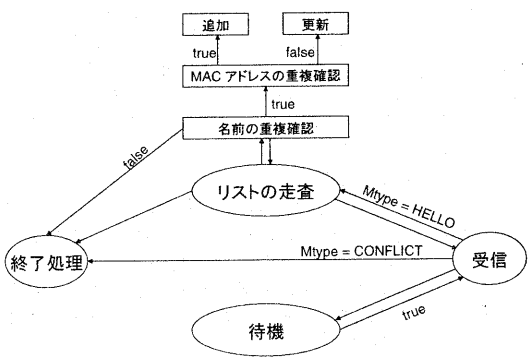

図 5　名前保存モジュール

ている手順に従ってその内容を保存する．パケットを受信するたびにそのノードの情報を更新することによって，他のノードの IP アドレスや名前の変化に対応する．名前保存モジュールに求められる一番重要なことは，ネットワーク内で衝突した名前を解決している間もユーザプログラムに対して迅速な名前解決サービスを提供することである．そこでこのモジュールは，受け取った RANR パケットを検査して名前の重複があった場合，最初に受け取った方のリストを保持しその新たなパケットに関しては何もしない．そうすることで自ノード内の名前の一貫性を保つことができる．もしネットワークで名前衝突が解決された後に，そのエントリの名前が変更された場合は再びリストを更新することになる．この名前保存モジュールは主にユーザプログラムのためのものであり，RANR での名前解決には直接関わらないため，このモジュールが無くても RANR は動作する．

5. RANR の評価

本稿では，RANR を FreeBSD3.3R[10] で動作する DSR 上に実装し評価した．本章では，まず MANET の構築に用いた DSR の概要を述べ，次に RANR の評価について述べる．

5.1 DSR の概要

DSR は CMU の MONARCH プロジェクトにおいて開発された MANET におけるルーティングプロトコルの一つであり，FreeBSD3.3R 上での実装が公開されている．本研究では RANR の実装，評価を行う MANET 環境として DSR を利用した．

5.2 RANR の評価

本節では，DSR を用いて実際に構築した MANET 上における RANR の性能評価の結果を述べる．測定環境としては RANR を組み込んだ 8 台の無線通信端末を用意し，図 6 に示されているようなすべてのノードが 1 ホップ内で通信できるネットワークを構築した．8 台のノードの平均 CPU 周波数は 291.46 MHz であり，無線通信媒体には 802.11b[11] 規格の MELCO BUFFALO 無線 LAN カードを使用した．

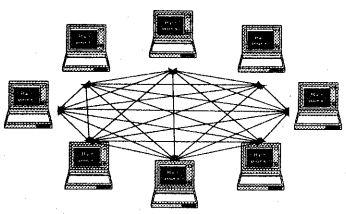

図 6　測定環境

評価項目は，以下の二つである．

- RANR のオーバヘッド
- ネットワーク内における名前衝突の解決に要する時間

なお，この評価はネットワーク内のノード数と HELLO パケットの送信間隔の二つのパラメータを変化させて行った．

図 7 は RANR の名前解決のためのパケットによるオーバヘッドの測定結果を示している．この測定は，名前の衝突は起こさず，HELLO メッセージを送る間隔を 3 秒に固定し，ノード数を変化させて行った．台数に比例してオーバヘッドがかかり，最大 8 台の時は 277Byte/sec となった．3 台以上になると，1 台増えるごとに約 20 から 30%オーバヘッドが増加する．

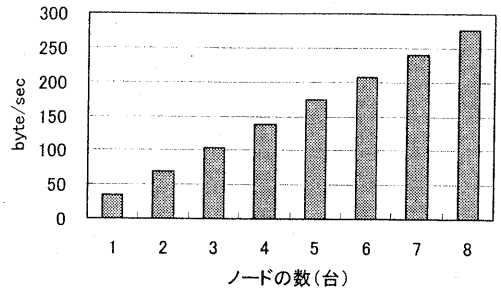

図 7　ノード数の変化による RANR のオーバヘッド

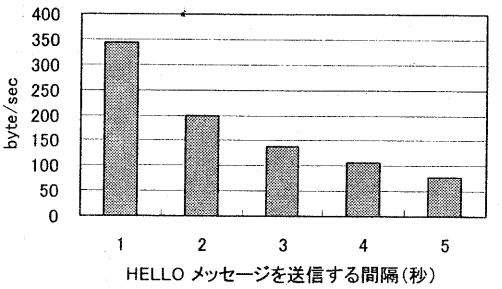

図 8　HELLO メッセージ送信間隔の変化による RANR のオーバヘッド

図 8 は，同じ測定を HELLO メッセージを送る間隔を変えて行った結果である．ノード数は 4 台に固定した．1 から 3 秒までは，1 秒間隔を伸ばすごとにオーバヘッドも 30%以上減少した．3 から 5 秒の間ではさほど変化が見られないため，このネットワークでは 3 秒程度が適当な値であると思われる．

次に RANR で名前解決のために要する時間を評価した．これはネットワーク内のすべてのノードの名前を衝突させて，解決にどのくらいかかるか測定したものである．本来は，衝突が検出された時点でユーザに新しい名前を要求するべきであるが，ユーザが操作する時間を含めずにシステムの性能を測定するため，予め複数の名前を候補として RANR に与え，名前の変更時にそれらの名前を順次用いた．またネットワーク内のすべてのノードが一斉に RANR による名前解決を開始するようにした．図 9 と図 10 は 10 回の測定結果の平均値と標準偏差値を示している．

図 9 は HELLO メッセージを送る間隔は 3 秒に固定し，ノード数を変化させて測定を行った結果を示している．3 台から 8 台の間では，名前解決に要する時間の差は 1 秒以内に収まっており，最大 8 台の場合でも 2.5 秒以内に解決が終了する．

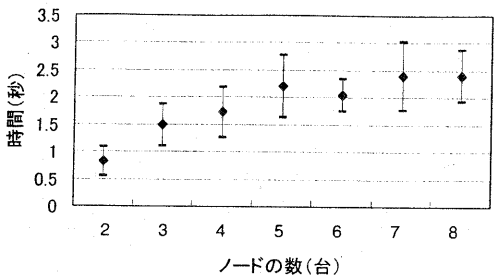

図 9 ノード数を変化させた場合の名前解決に要する時間

また，図 10 はノード数を 4 台に固定し HELLO メッセージを送信する間隔を変化させて名前解決にかかる時間を測定した結果である．短い間隔で HELLO メッセージを送る方が名前解決が早く終了することが分かる．これは，HELLO メッセージを受け取る回数が増えるとそれに比例して CONFLICT メッセージを出す回数も増えるためである．しかし，1 秒から 4 秒の間では名前解決にかかる時間の差は 1 秒以内に収まっており，さほど変わらない．そのため，図 7 とのトレードオフを考慮すると HELLO メッセージを送信する間隔はこの実験ネットワークでは 3 秒程度が適当である．3 秒より間隔を狭めてもそのオーバヘッドに対して名前解決に要する時間が改善されないためである．

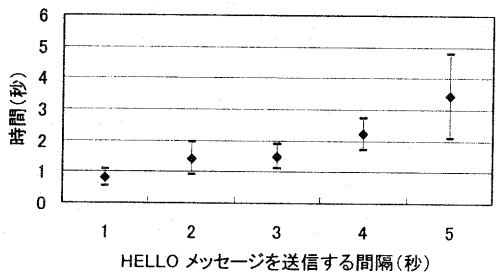

図 10 HELLO メッセージ送信間隔を変化させた場合の名前解決に要する時間

これらの測定結果は RANR がノード数 8 台のネットワークでそれほど大きなオーバヘッドをかけることなく，現実的な時間の範囲内で名前解決を実現していることを示している．また，RANR のスケーラビリティを 8 台まで確認した．さらに，HELLO メッセージを送信する間隔を調整することでネットワークにかかる負荷を調整できることが分かった．また，その間隔の短縮化が，生じる負荷と比較して必ずしも名前解決時間の短縮に結びつかないことも考察できる．ネットワークの規模に応じてこの値を調整し RANR を有効に動作させることができるが，比較的小規模なネットワークでは 3 秒程度の間隔が適当であると思われる．

6. おわりに

本稿では，MANET における名前解決の問題点を指摘し，その要件を分析した．これをふまえて，我々が想定する MANET 環境に最も適した名前解決機構として RANR を提案し，実装・評価を行った．RANR は完全分散型の名前解決機構であり，MANET 特有の，動的なトポロジ変化や通信リンク品質の不安定さ，IP アドレスの変化にも柔軟に対応する．また RANR では，各ノードが独自に任意の名前を付けることが可能であり，ネットワーク内でその名前が衝突した場合にも動的に解決する．また，既存のプログラムが RANR により割り当てられた名前を利用することも可能である．

評価実験の結果から，8 台のノードで構築されたネットワークにおいても，RANR が適切に名前解決を行えることを示した．具体的には，名前解決の際の RANR によるパケットオーバヘッドは平均 277Byte/sec であり，3 秒間隔の HELLO メッセージを用いた場合にはネットワーク内における名前衝突の解決に平均 2.5 秒の待ち時間を要した．これらの結果から，8 台で構築されたアドホックネットワークにおいては 3 秒間隔の HELLO メッセージの送信が適当であることを示した．本稿の評価環境において，RANR はオーバヘッドを小さく抑える努力をしつつ，現実的な時間内での迅速な名前解決を実現した．

参考文献

1) The MONARCH Project at Carnegie Mellon University, http://www.monarch.cs.cmu.edu/.
2) Johnson, D., Maltz, D., Hu, Y. and Jetcheva, J.: The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks, IETF Internet-Draft [Work in Progress] (2001).
3) Perkins, C., Royer, E. and Das, S.: Ad Hoc On Demand Distance Vector (AODV) Routing, IETF Internet-Draft [Work in Progress] (Jan.2002).
4) V. D. Park and M. S. Corson.: A highly adaptive distributed routing algorithm for mobile wireless networks, IEEE Infocom, 1997.
5) Albitz, P. and Liu, C., DNS and BIND, O'Reilly & Associates, Inc., Sebastopol, CA, 1994.
6) Vixie, P. (ed), Thompson, S., Rekhter, Y., Bound, J.: RFC-2136: Dynamic Updates in the Domain Name System (DNS UPDATE). (1997).
7) R. Droms, "Dynamic Host Configuration Protocol." RFC 1541, October 1993.
8) Zero Configuration Networking, http://www.ietf.org/html.charters/zeroconf-charter.html.
9) "MANETconf: Configuration of Hosts in a Mobile Ad Hoc Network", due to appear in the proceedings of Infocom 2002.
10) FreeBSD Project, http://www.freebsd.org.
11) IEEE Local and Metropolitan Area Network Standards Committee. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, IEEE Std 802.11-1997.
12) J. Lilley. Scalability in an Intentional Naming System. Master of Engineering Thesis, Massachusetts Institute of Technology, June 2000.