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

2006－OS－103 (3)
2006／7／31

DARTを用いたアドホックネットワークのアドレス空間均等化手法

徳永達也　田頭茂明　藤田聡

本稿では，大規模アドホックネットワーク向けに設計されたルーティングプロトコル DART (Dynamic Address RouTing) において，アドレス空間の均等化手法を提案する．DART は，動的な階層型アドレッシング手法とシンプルなルーティング手法を提供することにより，大規模ネットワークに対しても効率の良いルーティングを実現している．しかしながら，各ノードがアドレス空間を自律的に割り当てるために，ノード間で割り当てられるアドレス空間に偏りが発生する問題がある．そこで，アドレス空間の偏りを判断し，アドレスの再割り当てをおこなうことでアドレス空間を均等化する手法を提案する．提案手法をシミュレーションにより評価した．その結果，均等化の効果により，提案手法では従来手法よりも経路長を約 10%短くでき，また，経路表のサイズも減少することができた．

An Equalization Technique for Balanced Addressing in Adhoc Networks Using DART

Tatsuya Tokunaga　Shigeaki Tagashira　Satoshi Fujita

In this paper, we propose an equalization scheme of address spaces for DART (Dynamic Address RouTing) that is a routing protocol designed for a large scale adhoc network. In DART, the improvement of the scalability is achieved by using two main techniques; i.e., a dynamic and hierarchical addressing and a simplified routing. However, autonomous address allocation in each node causes the imbalance of address spaces assigned to nodes, which leads to significant performance degradation. In order to solve the problem, the proposed method realizes a re-addressing technique for the imbalance of address spaces. The effect of the proposed method is evaluated by simulation. The result indicates that by the effect of equalization, the proposed method shortens the path length by about 10% compared with DART protocol and reduces the size of routing tables.

1　はじめに

近年，物理的な場所に制約を受けないでネットワークを構築できる無線通信が，ユビキタス環境における基盤技術として注目を集めている．無線通信における重要な技術のひとつとして，アクセスポイントを必要としないネットワークの構築方法である，アドホックネットワークの研究が盛んに行われている．アドホックネットワークは，無線通信とルーティング機能を備えた端末 (ノード) によって構成されるネットワークのことで，通信範囲外のノードとは，他のノードがパケットを中継することで，そのノードとの通信を可能にしている．

アドホックネットワークでは，ノードの移動によりネットワークの構造が動的に変化するために，効率のよい安定した経路の発見手法が必要となる．このために，近年では数多くルーティングプロトコルが提案されている．アドホックネットワークでのルーティングプロトコルは，大きく分けてリアクティブ型とプロアクティブ型の 2 種類に分けることができる．リアクティブ型は，通信要求があった時に経路探索を行って通信経路を決定する方式で，動的なネットワークに対応できるが，経路を探索するまでに時間が必要である．プロアクティブ型では，定期的に経路情報を交換して，各ノードが経路表を保持することで，通信要求があると即座に通信を開始できるが，通信がない場合においても経路情報を交換する必要がある．

既存のルーティングプロトコルの多くは，数百ノード以下のネットワークでしか効率的に機能しないことが知られている．この理由としては，リアクティブ型では，経路の発見にフラッディングをベースにしていることから，

広島大学大学院工学研究科情報工学専攻
〒 739-8527 東広島市鏡山 1-4-1
Graduate School of Engineering, Hiroshima University
Kagamiyama 1-4-1, Higashi-Hiroshima, 739-8527 Japan

大規模なネットワークでは経路の発見時に，より多くのコストと時間が必要であり，また，プロアクティブ型では，ノード数の増加により経路表のサイズが大きくなり，各ノードは経路表を維持／交換するのにより多くの資源とコストが必要になるからである．このために，大規模なネットワークにも適用可能なプロトコルとして，プロアクティブ型の階層型ルーティングが提案されている．階層型ルーティングは，各ノードが保持する経路情報を削減し，経路情報を維持するのに必要な負荷を軽減させる手法である．階層型ルーティングには，DART[1]，HSR[2]，Tribe[3]などがある．DART (Dynamic Address RouTing) は，動的な階層型アドレッシング手法とシンプルなルーティング手法を提供することにより，大規模ネットワークに対しても効率の良いルーティングを実現している．しかしながら，各ノードがアドレス空間を自律的に割り当てるために，ノード間で割り当てられるアドレス空間に偏りが発生する問題がある．

本稿では，このDARTにおける問題を解決するためにアドレス割り当ての均等化手法を提案する．提案手法では，アドレス空間の偏りを判断し，アドレスの再割り当てをおこなうことでアドレス空間を均等化する．本研究では均等化の効果について，従来の手法と比較して評価を行った．その結果，均等化の効果により，提案手法では従来手法よりも経路長を約10%短くする事ができ，また，経路表サイズの減少により，経路維持にかかるオーバーヘッドを減らすことが出来た．

2 DART

2.1 概要

アドホックネットワークにおいて，プロアクティブ型のルーティングプロトコルは，ノード数の増加により経路表のサイズが大きくなる問題がある．このために，DARTではプロアクティブ型の階層型ルーティングプロトコルを提案し，経路表のサイズを抑えつつ効率的なルーティングを実現している．DARTのルーティングプロトコルは，アドレスの動的な割り当て，ルーティング，ノードルックアップの三つの機能から構成される．

DARTにおける各ノードは，識別子 (ID) と，ネットワークアドレスを保持している．IDは，固有の番号で，ネットワーク内に存在している間は不変である．ネットワークアドレスは，ネットワーク上の論理的な位置情報を示し，ノードの移動と共に動的に変化する．DARTにおいては，このネットワークアドレスを用いて，ノードへのルーティングが行われることなる．

ネットワークアドレスについて詳しく説明する．DARTにおけるネットワークアドレスは，アドレスの集合を二分木（以下，アドレス木と呼ぶ）で表わしたものをベースにしている．例えば，3ビットのアドレス木は，図1に

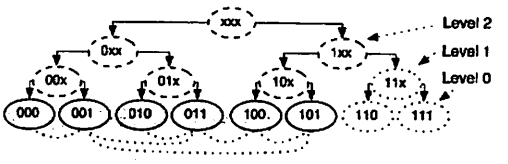


図 1: 3ビットのアドレス木．

示すようになる．アドレス木の葉に相当するアドレスが，実際にノードに割り当てられるアドレスで，ℓビットの二進数 $a_{\ell-1},\ldots,a_0$ で表わされる．葉以外の頂点は，アドレス空間と呼ばれ，その頂点を根とする部分木に含まれる葉のアドレスの集合を表す．例えば，図1において，アドレス空間 [1xx] は，アドレス [100] から [111] までの4つのアドレスの集合を表すことになり，そのアドレス空間のサイズは，4となる．

アドレス木において，高さℓの木は，レベルℓの部分木と呼ばれ，兄弟関係にあるレベルℓの部分木は，レベルℓのシブリング（sibling）部分木と呼ばれる．各ノードは，ルーティングのための経路表として，各レベルのシブリング部分木のアドレスを持つノードの情報を持つことになる．例えば，アドレス空間 [100] のシブリング部分木は，[101]，[11x]，[0xx] となる．また，アドレス [100] を持つノードは，アドレス [101](レベル 1)，[011](レベル 3) を持つノードへのエントリを経路表に保持している．アドレス [11x] を持つノードは無いので，レベル2のエントリは空である．

次節からは，このネットワークアドレスを用いた，動的なアドレス割り当て，ルーティング，ノードルックアップについて詳しく説明していく．

2.2 アドレスの動的な割り当て

DARTにおけるアドレスの割り当ては，新しいノードがネットワークに参加するときに実施される．ノードは，アドレス又はアドレス空間が動的に割り当てられる．アドレス空間が割り当てられたノードは，そのアドレス空間に含まれるアドレスの一つを自身のアドレスとし，その他のアドレスは，隣接ノードに割り当てるために保持しておく．

図2は，DARTのアドレス割り当ての様子を表わしている．図2では，右図で示すように，まずノードAが参加しており，その後，ノードB，ノードC，ノードDが参加する状況を想定している．左図は，そのような状況におけるアドレス木の様子を示している．最初，ノードAは，アドレス [000] と3ビットのアドレス空間全体を保持している．このノードのことをアドレス木の根と呼ぶことにする．次に，ノードBが参加しノードAと接続した

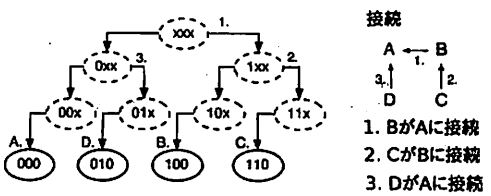


図 2: アドレス割り当ての例.

とき，ノード A は自身の経路表の中でレベルが最も高く，かつ未割り当てなアドレス又はアドレス空間を，ノード B に割り当てることになる．割り当て後，経路表上の割り当てたアドレス又はアドレス空間のエントリには，割り当て済みのフラグが記述される．例えば，図 2 の場合では，ノード B はアドレス空間 [1xx] が割り当てられることになる．もし，参加するノードが，二つ以上の隣接ノードからアドレスの割り当てが実施された場合では，割り当てられるアドレス空間のサイズが最も大きいものを選択することになる．

ノードが，ネットワークから離脱するときには，アドレスを受け取った隣接ノードにアドレス空間を返還する．

2.3 ルーティング

ノードがルーティングを行うとき，まず最初に，目的地のアドレスが所属するシブリング部分木のレベルを決定する．すなわち，ノードは自身のアドレスと目的地のアドレスを最上位ビットから比較し，最初に異なる値になったビットから，シブリング部分木のレベルを決定する．例えば，図 2 において，アドレス [000] から [100] へルーティングを行う場合，最上位ビットが異なるので，[000] から見て [100] が所属するシブリング部分木のレベルが 2([10x]) であると分かる．各ノードの経路表は，各レベルのシブリング部分木内の一つのノードへの経路が記されているので，レベル 2 のエントリを参照して，次ホップを決定する．パケットがシブリング内のノードに到達すると，さらにそのノードが自身のアドレスと目的地のアドレスを比較して次のシブリング部分木を決定する．このときのシブリング部分木のレベルは必ずその前のレベルよりも小さくなる．この手順はパケットが目的地に到達するまで繰り返される．

このようにルーティングをすることで，以下のような利点がある．

1. 各ノードが保持する経路表のサイズを大幅に減らすことができる．
2. 隣接ノードと交換する経路表のサイズも同様に減らすことができる．
3. トポロジーの変化が近隣のノードにのみ影響を及ぼし，遠くのノードは経路表を書き換える必要が無い．

経路表の更新は，隣接ノードから経路表情報を受け取ったときに行われる．経路表は，宛先アドレス (dest)，宛先への次ホップ (next)，宛先までのホップ数 (hops) からなる．宛先アドレスは各レベルのシブリングのアドレスを示す．図 3 は，アドレスが [0111] のノードとアドレスが [0100] のノードの経路表の更新を図で示したものである．

二つのノードのアドレスの共通するプレフィックスを k ビットとする．このとき，二つのノードのレベル ℓ からレベル $(\ell-k+1)$ のシブリング部分木が共通なので，ルーティングテーブルの交換はレベル ℓ からレベル $(\ell-k+1)$ までのエントリと，レベル $(\ell-k)$ のみに関係がある．よって，経路表の更新は以下のように行われる．

1. レベル ℓ からレベル $(\ell-k+1)$ まで

   このレベルはアドレスのプレフィックスが同じなので，二つの経路表の宛先アドレスが等しくなる．図 3 では，レベル 4 とレベル 3 の宛先アドレスに相当する．更新メッセージを受け取ったノードは，各レベルのホップ数を相手のテーブルと比較して，もし相手のホップ数+1 が自分のホップ数より小さければ，次ホップを相手のアドレスにしてホップ数を更新する．

2. レベル $(\ell-k)$

   二つのノードは隣接するので，レベル $\ell-k$ のホップ数は 1 となる．もし，自身の経路表のホップ数が 2 以上であれば，次ホップを相手のアドレスにしてホップ数を 1 とする．図 3 の場合はレベル 2 の宛先アドレスに相当する．

3. レベル $(\ell-k-1)$ からレベル 0 まで

   このレベルはアドレスのプレフィックスが異なるので，二つのテーブルの宛先アドレスが異なる．よって，このレベルのエントリは更新されない．

2.4 ノードルックアップ

DART では，通信するノードの現在のアドレスを知るために，分散ノードルックアップ表の利用が考えられている．これはルックアップ表を構成する $\langle ID$, ネットワークアドレス$\rangle$ という組を，各ノードが分散して管理する手法である．ネットワークに参加するノードは，新しく得たネットワークアドレスと自身の ID を，分散ノードルックアップ表に書き込む．ノードが通信を行うときには，ルックアップ表を用いて，ID から送信先の現在のネットワークアドレスを獲得する．

具体的には，各ノードが管理するエントリを決定するために，ピア間でユニークなハッシュ関数を利用する．例えば，識別子 ID_1 のノードがあり，そのネットワークアド

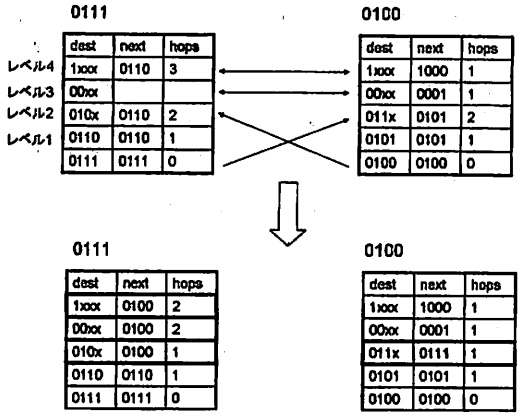
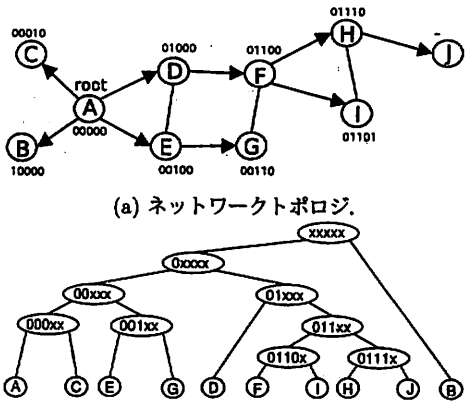


図 3: テーブルの更新.



(a) ネットワークトポロジ.

(b) アドレス木.

図 4: 不均等なアドレス割り当ての例.

レスが [010] だった場合において，ハッシュ値 $hash(ID_1)$ のアドレスを持つノードが，エントリ $(ID_1, 010)$ を管理する．アドレス $hash(ID_1)$ を持つノードが存在しない場合は，最も近いアドレス (共通プレフィックスが大きいアドレス) がエントリを保持することになる．

2.5 アドレス割り当ての偏り

DART では，アドレス木の根を中心にして，アドレスの割り当てを行いネットワークを形成する．このために，根がネットワークの (物理的に) 中心にあり，その根を中心に均等にノードが参加していく場合では，DART を利用することで，非常に良い性能が得られる．しかしながら，根の位置がネットワーク内の中心とは離れた場所にある場合や，根から偏った方向にノードが参加していく場合などでは，DART を用いたとしても期待した性能が得られない．このような状況は，アドホックネットワークでは一般的であり，DART において解決すべき問題のひとつとして考えられる．

例えば，図 4(a) において，根となるノードを A としてアドレス割り当てを行った結果を示している．矢印はノードの親子関係を表わし，アドレスを割り当てたノードを親，割り当てられたノードを子とする．これをアドレス木で表わすと図 4(b) のようになる．アドレス木の葉 (ノード) の深さは，そのノードがアドレスを持つのに必要なビット数となる．ノード数が N のとき，本来ならば高さ $\lceil \log N \rceil$ のツリーで全てのノードにアドレスを割り当てることが可能だが，アドレス割り当てに偏りが生じると余分なビット数が必要となる．

我々はこの現象がシステムに与える影響を確かめるために，シミュレーションによる実験を行った．ノードが一様に分布したネットワークにおいて，ネットワークの中央にあるノードを根として，アドレス割り当てがほぼ均等に行われた場合と，ネットワークの端にあるノードを根として，アドレス割り当てが特定のプレフィックスに偏った場合について全体の経路表のサイズと通信で必要なホップ数の平均を求めて比較した．ノード数は 500 から 4000 で，ランダムに選んだ 1000 組のノード間の通信ホップ数を求めた．

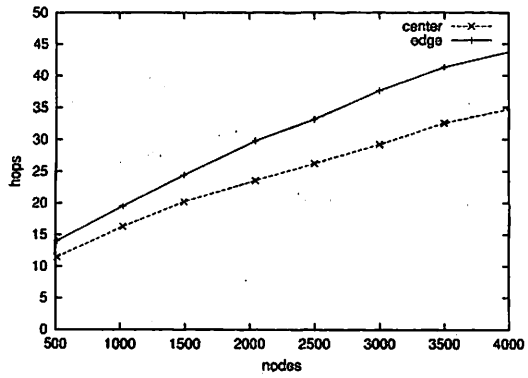
図 5(a) はホップ数，図 5(b) は経路表のサイズ (エントリ数) の結果を示している．アドレス割り当てに偏りが発生した場合，発生しない場合と比較してホップ数で約 20〜30%，経路表サイズで約 30〜90%の増加が見られる．また，ノード数が増えるほどその影響が大きくなっていることがわかる．このように，根の位置によるアドレス割り当ての偏りが，性能におよぼす影響は無視できないほど存在するために，アドレス割り当てを均等化する手法について考察していく．

3 提案手法

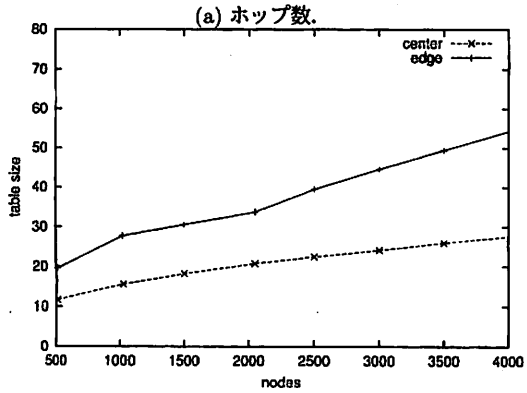
3.1 概要

提案手法では，根がアドレス割り当ての偏りを発見すると，根の権限を他のノードに移動させ，それに伴い幾つかのノードにアドレスの再割り当てを行うことによりアドレス割り当ての整合性を保ちつつ均等化を図る．

この手法による問題点は，アドレスの再割り当て後の新しい経路情報がネットワーク全体に行き届くのに時間がかかることである．これを解決するため，根の移動期間中はノードは古いアドレスと新しいアドレスを同時に持つようにし，新しい経路情報が行き届くのに十分な時間が経過するまでは，古いアドレスを用いて通信し，その後は新しいアドレスを用いて通信できるようにする．



(a) ホップ数.



(b) ルーティングテーブルサイズ（エントリ数）.

図 5: 根の位置による性能比較.

3.2 偏りの発見

根がアドレス割り当ての偏りを調べるには，各ノードが保持する経路表のサイズの情報を利用する．あるノードが持つ経路表のサイズは，アドレス木で表わすとそのノードの深さを表していることになる．サイズが大きいほど，ID に多くのビット数が使われているため，アドレス割り当てが偏っていると言える．

サイズの情報を集めるため，各ノードは経路表の交換時に経路表のサイズを親に伝える．さらに，親は子孫の最大サイズを自身の親に伝えることになる．この操作を根に到着するまで繰り返す．最終的に根のノードに各レベルのシブリングの最大サイズの情報が集まってくることになる．根はこれらのシブリングから受け取った経路表のサイズの最大値と最小値の差をとり，その差 S がある閾値 T 以上であれば，アドレス割り当てが偏っていると判断して根の移動を開始することになる．

3.3 根の移動

根の移動は，まずアドレスを再割り当てしなければならない根以外のノードの有無を判定する．もし，再割り当てするノードが存在する場合には，それらのノードの再割り当てを実施する．その後，根は，次の根となるノードを適切に選択し，そのノードに対して根の移動を行う．

次の根の選択は，現在の根から見て，最もアドレス割り当てが偏っているシブリングに $(S-T)/2$ ホップ移動した先のノードとする．根を 1 ホップ移動するごとに，エントリの差 S は大体 2 ずつ小さくなるので，これにより根の移動後のエントリ数の差を閾値以下まで小さくすることができる．

3.3.1 根以外のノードの移動手順

アドレスを再割り当てする必要がある根以外のノードの有無は，根は各シブリング内のノードと隣接しなければならないという制約から，根の移動により隣接しなくなるシブリングのノードの存在で判定することができる．アドレスの再割り当てはそのシブリング以下のノードに対して行われる．

このようなノードが存在する場合には，以下のステップを実行することにより，それらノードのアドレスを再割り当てすることになる．以下では，移動前の根ノードを r_{old} と示し，移動後の根ノードを r_{new} と示すことにする．また，再割り当てする必要がある根以外のノード集合を A と表現する．

ステップ 1 アドレスのロック

r_{new} は，再割り当てされるノードの古いアドレスを，これから参加するノードに割り当てないようにする．

ステップ 2 アドレスの再割り当て

r_{old} は，ノード集合 A に対して，再割り当ての指示を出す．指示を受け取ったノード集合 A は，隣接ノードに AREQ (address request) メッセージを送る．AREQ メッセージを受け取ったノードは，アドレス空間の一部を委譲する AREP (address reply) メッセージを返す．アドレスを再割り当てされたノードはアドレスの解放命令があるまで，新旧 2 つのアドレスを使用する．新しいアドレスの割り当てが終了したノードは，r_{new} に再割り当て完了の ACK を送る．

ステップ 3 根の移動

再割り当て完了の ACK をすべて受け取った r_{old} は，3.3.2 節の根の移動手順を実行する．

ステップ 4 古いアドレスの解放

一定期間 t_r が経過した後，ノード集合 A は古いアドレスを破棄し，r_{new} に ACK を送る．

ステップ5 アドレスのロック解除

ノード集合 A のすべてのノードからACKを受け取った r_{new} は，ステップ1でロックしたそのシブリングのアドレスのロックを解放し，別のノードに割り当てることが出来るようにする．

3.3.2 根の移動手順

r_{old} から，r_{new} に根を移動する手順を以下に示す．

ステップ1 根の権限の譲渡

r_{old} から，r_{new} に根の権限を渡け与えるために，メッセージ RAUT(root authority) を送る．

ステップ2 アドレス0の割り当て

RAUT を受け取った r_{new} は，アドレス0をつける．このとき，r_{new} が付けていた古いアドレスと，新しいアドレス（アドレス0）の両方をつけることになる．その後，r_{old} へのACKを返す．

ステップ3 新しいアドレスの割り当て

r_{old} は，r_{new} からのACKを受け取ると，隣接ノードに，AREQ メッセージを送信し，新しいアドレスを受け取る．このとき，r_{old} は経路表のサイズが最も小さい隣接ノードに AREQ メッセージを送る．r_{old} は，受け取った新しいアドレスと古いアドレス（すなわち，アドレス0）の両方をつけることになる．

ステップ4 古いアドレスの解放

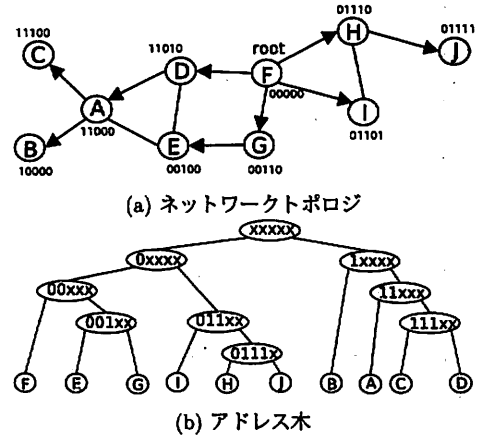
一定期間 t_r が経過した後，r_{old} と r_{new} の古いアドレスを解放する．

図4の例でアドレス空間の均等化を行った場合，均等化後は図6のようになる．各ノードが保持するアドレス空間の差が小さくなり，均等化が行われていることがわかる．

3.4 シーケンス番号を用いたルーティング

3.3.2節で述べたように，アドレスの再割り当ての期間において，再割り当ての対象となるノードは新旧2つのアドレスを持つことになる．提案手法では，このような状況において，新旧2つのアドレスのどちらのアドレスを用いたとしても，目的とノードと通信できるように，DART のルーティング手法を拡張している．拡張のポイントは，経路表に保持されるエントリにシーケンス番号を付与し，再割り当て前の経路と再割り当て後の経路とを区別し，同時に利用できるようにすることである．

シーケンス番号は，ネットワーク上でグローバルな変数とし，2.4節で述べた仕組みを利用してノード間で共有する．新しいアドレスが再割り当てするノードに行き届いた後，新しい根がシーケンス番号を1インクリメントする．再割り当てを行うノードは，古いアドレスを含む経路表に加えて，シーケンス番号をインクリメントした新しいアドレスを含む経路表を作成し，それら2つの経路表を他のノードに伝搬することになる．古いシーケンス番号の経路表は，古いアドレスを解放した後，各ノードで自動的に削除されることになる．



(a) ネットワークトポロジ

(b) アドレス木

図 6: 均等化後のアドレス割り当ての例．

この新旧アドレスを含む経路表の交換について述べる．図7に示すように，アドレスの再割り当てが行われたノードAは，再割り当て前 (seq no.99) の古い経路表と，再割り当て後 (seq no.100) の新しい経路表を持っているとする．ノードBとCは，再割り当てが行われなかったノードである．すなわち，古い経路表しか持たないノードである．新しい経路情報がネットワーク全体に行き渡るまでの間，ノード間の経路表の交換は，(1) 古い経路表のみを持つノード間の交換，(2) 新旧2つの経路表を持つノード間の交換，(3) 古い経路表のみを持つノードと，新旧2つの経路表を持つノードとの交換の3通りが考えられる．このうち (1) と (2) の場合は，同じシーケンス番号の経路表間で，2.3節で述べた方法で経路情報の交換が行われる．(3) の場合として，例えば図7の左図のように，新旧2つの経路表を持つノードAが，古い経路表のみを持つノードCに経路情報の更新メッセージを送った状況を考える．この場合，ノードCのアドレスは新しい経路表でも有効なので，ノードCは古い経路表をコピーして新しい経路表を作成し，同じシーケンス番号の経路表間で，2.3節で述べた方法で経路情報の交換が行われる．また，古い経路表のみを持つノードBが，新旧2つの経路表を持つノードCに更新メッセージを送る状況では，ノードBのアドレスは新しい経路表でも有効なので，ノードCは受け取った経路表を用いて新旧2つの経路表を更新する．

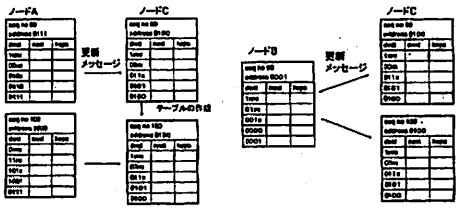


図 7: シーケンス番号を用いた経路表の更新.

以上の操作を繰り返すことで，ネットワーク内のノードに新旧 2 つの経路表が作成されることになる．

データを送信するノードは，現在のシーケンス番号を獲得し，データを構成するすべてのパケットにそのシーケンス番号を付けて送信する．各ノードはパケットのシーケンス番号と，経路表の送信先アドレスのシーケンス番号が一致するエントリを用いてルーティングを行う．

4 実験

4.1 実験方法

提案手法の効果についてシミュレーションを行い，平均ルーティングテーブルサイズ，平均ホップ数，オーバーヘッドについて評価する．実験環境としては，ノードはシミュレーション空間内に一様にばらまき，根は最初にネットワーク上の端にあるノードを選択した．また，本シミュレーションでは，ノードは移動しないものとする．具体的なパラメータはとしては，ノード数は 64 から 1024 までを変化させ，アドレスで用いたビット数は 64，各ノードの通信範囲を 200m とし，根を移動するための閾値を logN とした．

4.2 実験結果

4.2.1 テーブルサイズ

図 8 は，ノード数を変化させたときの各ノードの経路表のサイズの平均を求めたものである．$\log N$ と比較した場合，従来の DART は 70-80%の増加が見られるが提案手法は 20-30%の増加にとどまっている．この理由は，従来手法ではアドレスの偏りによりサイズが増加していたが，提案手法では，均等化により，サイズをより理想に近づけることができたことによるものだと考えられる．

4.2.2 ホップ数

図 9 は，ネットワーク内の二つのノードで通信を行った際のホップ数の平均を求めたものである．DART は最短経路を保証する手法ではないので，ホップ数の増加 (path stretch) が生じる．図では，AODV との比較も載せている．AODV[4] は，リアクティブ型プロトコルの一つで，通信要求があると経路要求メッセージをブロードキャストして経路探索を行い，最短経路を選択する手法である．AODV と経路長を比較した場合，従来手法は約 35%の経路の増加となるが，提案手法は約 20%の増加に抑える事ができていることがわかる．この理由は，4.2.1 節で述べた理由と同じで，提案手法の根の移動によりアドレス空間の偏りが改善されたことによるものだと考えられる．

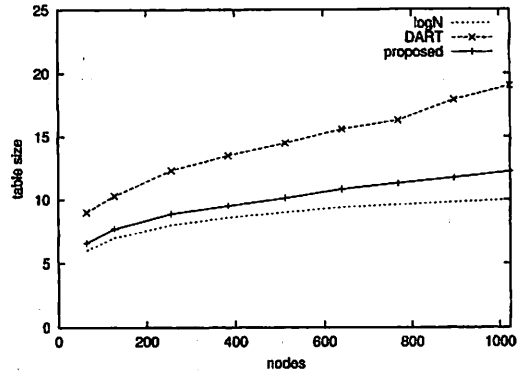


図 8: ルーティングテーブルサイズ（エントリ数）.

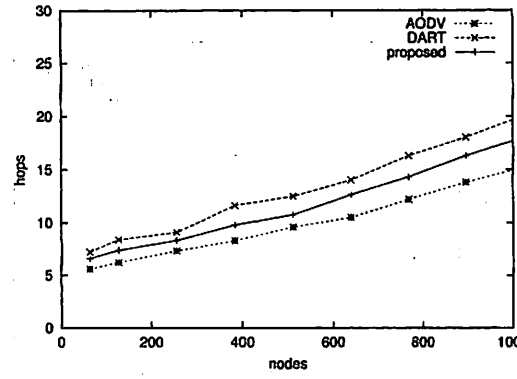


図 9: 平均ホップ数.

4.2.3 プロトコルオーバーヘッド

図 10 は，ノード数を変化させたときのプロトコルに必要なメッセージ量 (KB/s) を示している．フロー数を 100 に固定している．AODV は，経路探索にフラッディングを用いているため，ノード数の増加が深刻なメッセージ量を招いていることがわかる．DSDV[5] はプロアクティブ型のフラット型ルーティングを行うプロトコルで，メッセージ量はノード数に比例して増加する．DART は交換する経路表サイズが DSDV より大幅に少ないので，メッ

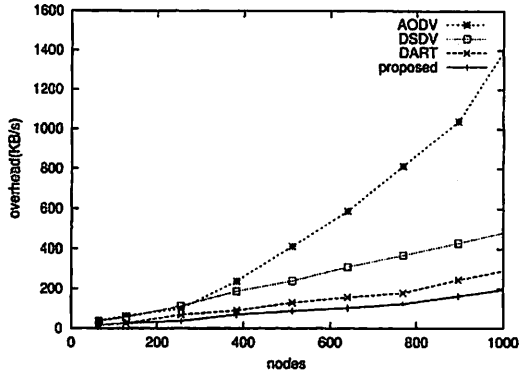


図 10: ノード数を変化させたときのメッセージ量.

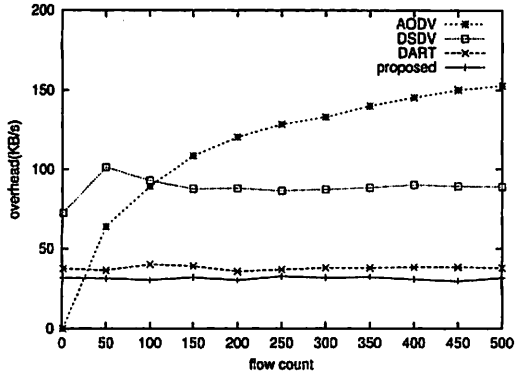


図 11: フロー数を変化させたときのメッセージ量.

セージ量が少ないことがわかる. 提案手法では, DART と比べてさらにメッセージ量を削減することができている. この理由は, 均等化により, 従来の DART よりもさらに経路表サイズを小さくすることでき, 経路表の交換に必要なメッセージ量を削減できているからである.

図 11 は, 通信のフロー数を変化させたときのプロトコルに必要なメッセージ量 (KB/s) を示している. この実験では, ノード数は 200 で固定している. AODV は, リアクティブ型のプロトコルであるために, 経路要求が無い場合 (フロー数 0) のメッセージ量は 0 である. しかし, 通信要求が増加するにつれてメッセージ量も増加することがわかる. DSDV や DART は, プロアクティブ型のプロトコルであるために, フロー数によらずメッセージ量は一定となる. 提案手法は従来手法よりメッセージ量を小さくする事が出来る. また, フロー数が 30 以上のときは, AODV よりメッセージ量を少なくなる.

5 おわりに

本研究では, DART の問題点について述べ, その問題点を解決する手法としてアドレス割り当ての均等化を提案した. DART は各ノードが自律的にアドレス割り当てを行っているため, ネットワーク全体の構造について把握できない. そこで, ネットワークの根のノードを基準としてネットワークのアドレス割り当ての偏りを判断し, 根の権限の移動とアドレスの再割り当てをおこなうことにより均等化を実現した. この手法により, アドレス割り当ての偏りが発生した場合に動的に対処してアドレス空間を効率的に利用できるようになった.

本研究では均等化の効果について, 従来の手法と比較して評価を行った. その結果, 提案手法では従来手法よりも経路長を約 10%短くする事ができた. また経路表サイズの減少により, 経路維持にかかるオーバーヘッドを減らすことが出来た. また他のプロトコルと比較しても, 優れた性能を発揮できた.

今後の課題としては, ノードのモビリティを考慮した上での適切な閾値の設定方法や, 根以外のノードでの局所的な均等化手法の提案等が挙げられる.

参考文献

[1] Jakob Eriksson, Michalis Faloutsos, and Srikanth Krishnamurthy. Scalable Ad Hoc Routing: The Case for Dynamic Addressing. In *Proc. of IEEE INFOCOM*, 2004.

[2] Guangyu Pei, Mario Gerla, Xiaoyan Hong, and Ching-Chuan Chiang. A wireless hierarchical routing protocol with groupe mobility. In *Proc. of WCNC*, 1999.

[3] Aline C. Viana, Marcelo D. de Amorim, Serge Fdida, and Jos F. de Rezende. Indirect routing using distributed location information. In *Proc. of ACM Mobile Networks Application, Special Issue on Mobile and Pervasive Computing*, 2003.

[4] Charles E. Perkins Elizebeth M. Belding-Royer and ian Chakeres. Ad hoc On-Demand Distance Vector Routing. *IEEE Workshop on Mobile Computing Systems and Applications*, pages 90–100, 1999.

[5] Charles Perkins and Pravin Bhagwat. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In *Proc. of the SIGCOMM 1994 Conference on Communications Architectures, Protocols and Applications*, 1994.