Vol.2010-MBL-55 No.17
2010/9/3

マルチホップ無線アクセス網 NerveNet 上への ドロネーオーバレイネットワークの 分散構成法の提案

大西真晶[†]　　井上真杉[†]

インフラ型マルチホップ無線ネットワーク NerveNet の基地局ノードはデータベースを備えており，地域に展開されることにより，ネットワーク，データベース，計算資源の情報システムを構成する主要な機構を地域に対して提供する．本研究提案では NerveNet のデータベース間の論理ネットワークのトポロジとして空間データ構造の一種でもあるドロネーグラフ状のネットワークであるドロネーオーバレイネットワークの利用を提案し，その利点とルーティングテーブルの分散生成法についても示す．

A proposal of the distributed Delaunay overlay network construction algorithm on NerveNet

Masaaki Ohnishi[†]　and Masugi Inoue[†]

The base stations of NerveNet which is an infrastructure type network is equipped with the database. The base stations placed in local area provides a network, databases, and CPUs to local community. And it is used as a telecommunications service platform on local community. In this study proposal, use of the Delaunay overlay network is proposed as topology of the logical network between the databases of NerveNet. And the distributed generating method of routing table is shown.

1. はじめに

我々が研究開発を進める NerveNet（図 1）は，主に地域社会での利用を想定した情報通信サービスプラットフォームである[1][2]．無線基地局を地域に固定的に設置し，近傍基地局間を有無線でマルチホップ接続することで，地域全体で管理されたデータ通信網を展開する．地域に閉じた LAN であるが，利用者にアクセス権限があれば特定の GW を介することでグローバルなネットワークへ接続することもできる．地域に存在する自治体，家庭，公共機関，企業などの様々なプレイヤが，この網にサーバを接続することができ，網内の端末に対して情報を公開することができる．

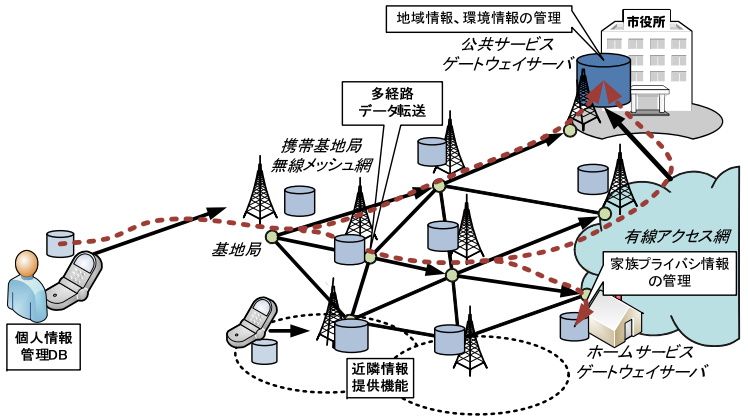

図 1　NerveNet

各基地局は L2 スイッチを用いてネットワークを構成するが，その時，多経路を選択可能とするように VLAN を多数設定する．これにより多経路を用いたデータ転送を可能とする．各基地局はデータベースを持ち端末の所在情報や認証情報などのネットワークを機能させるための制御情報を全基地局間で交換し管理する．更にプッシュ配信に用いるコンテンツ情報についても，コンテンツ情報そのものや，コンテンツ情報のネットワーク上での所在を各基地局間で共有する．また，基地局上にて任意のソフトウェアを走らせることもでき，通信状態，データベースに基づく様々な処理を実行できる．これにより基地局コンポーネントのみでネットワーク機構，データ管理機構，情報処理機構という情報通信サービスプラットフォームが必要とする主要機構を提供し，敷設コストと管理コストの低減を目指している．

NerveNet は平常時においては，地域住民への様々なサービスを行うために利用される．低料金，もしくは無料での地域特有の各種情報サービスの提供，地域に閉じ管理された未成年者の生活圏を十分にカバーする安全な情報ネットワークの提供と，これを用いた ICT リテラシ教育の支援環境提供．センサを利用した子供，老人などの見守りシステムの構築，既存のインフラの行き届かない広大な土地などの監視システムの構築，地域経済活性化の為の宣伝広告システム，取引システムなどに利用される．

[†] 独立行政法人 情報通信研究機構
National Institute of Information and Communications Technology

Vol.2010-MBL-55 No.17
2010/9/3

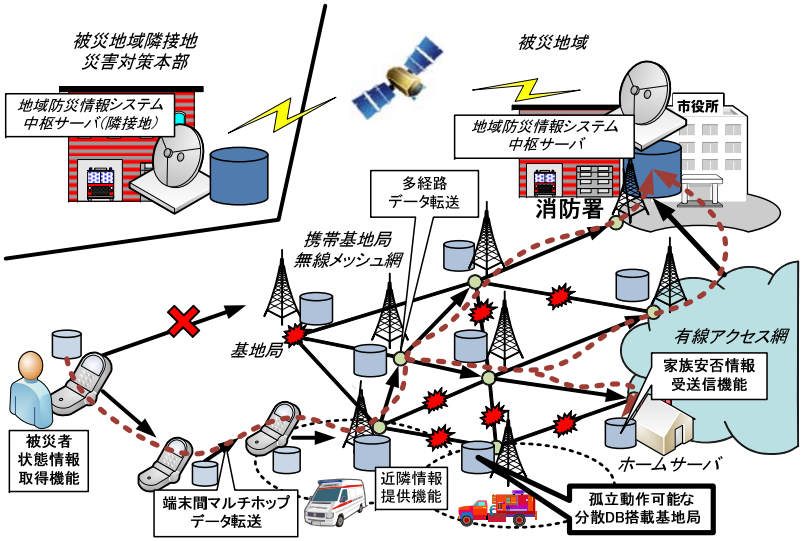

図 2　大規模災害時に NerveNet が果たす役割

更に NerveNet は大規模災害時にも完全動作可能な情報通信システムとしての役割も担う（図 2）．NerveNet はインフラ型のマルチホップネットワークであり，近傍接続のメッシュネットワークである．これに充電池や太陽電池パネルなどの独立給発電設備を搭載することにより，広域の大地震などが起こり，大規模な構造物の破壊と停電が起こった場合にも地域網としての全体的な接続と動作を保つことができる．これにより地震発生直後から安否確認，通報，被害把握などを行うことが可能となる．また基地局にデータベースが搭載されている為，万が一ネットワークが寸断されたとしても,そのデータベースを使用することによって非常時用の情報サービスを提供する．従来の通信網は，冗長化されてはいるが大規模な災害時に必ず動作することを目指していない．例えば 1995 年の兵庫県南部地震においては，被害の大きかった地域では電柱設備の 2 割以上が倒壊・破損し架空線を寸断しアクセス網が無力化した．また，停電が 2 時間で復旧したにも関わらず通信網の機能自体は 1 日以上復旧しなかった．これは網の管理が人の手を前提としたものであり，交通機関が寸断されるような大規模災害時には人の手によるシステムの再起動ができなかった為である[3].

NerveNet では自動網構築機構と自動管理機構を研究開発,搭載して平常時の管理コストの低減を目指す．同時に災害時のロバスト性を高める．様々なニーズに対する統合的なインフラとして機能させ社会全体のシステムコストの大幅な低減も目指す．本研究では NerveNet 上のデータベース間の論理ネットワークとしてドロネーオーバレイネットワーク[4]の迂回経路テーブルを含む経路テーブルを自律分散生成する手法を示す．このオーバーレイネットワークは分散的なデータ管理とデータ提供，ジオメトリックルーティングによる長距離データ転送実装の土台となるものであり重要である．適用可能先は NerveNet に限らずクラスタリングを行ないサブネットワークの連結する機構を組み込める実装を持つネットワーク全般である．２章にて NerveNet におけるドロネーオーバーレイネットワークの構成，３章にてドロネーオーバーレイネットワーク経路表の分散生成法，４章にて経路表の分散生成法シミュレータの試作，５章にて関連研究を示す.

2. NerveNet におけるドロネーオーバーレイネットワークの構成と役割

2.1 NerveNet とドロネーオーバレイネットワークの関係

NerveNet の各基地局は見通しが効きやすい近傍の基地局間で無線を主に使用した物理リンクを構成する．更に各基地局からネットワーク全体に対してツリー状の VLAN を設定する．NerveNet では，こうして構成された多数の VLAN のうち，その時点で使用可能なものを利用して基地局間の通信を行う．リンクが十分に安定的で大きな通信容量を保持していれば,この方法のみで NerveNet を構成することは十分に可能である．しかし，NerveNet は網の簡易な設置と全自動の網構築も目標としている．その為，ネットワークリンクが不安定で切断通知が大量に発生する場合や，なんらかの理由でリンクの通信容量が十分でない場合でも，その性能が許す限りでサービスを提供できなくてはならない．また大規模災害時には広域に渡ってバックボーンネットワークが破壊されることも考えられる．このような場合には各地域に敷設された NerveNet を乗り継ぎ長距離のデータ転送を行うことも考えられる．しかし，切断通知が頻出する無線リンクで大規模なネットワークを構成することは困難であり，我々が提案してきた従来的なネットワーク構成手法では対応できない可能性が高い．適切なクラスタリングによる制御信号流量の低減を行う必要がある．本提案におけるドロネーオーバレイネットワーク利用を NerveNet に適用した場合，各基地局は L2 ネットワーク資源消費の最低化を目指す場合で地理的に近傍な約 6 基地局を探索し，それらとの間で制御情報のやりとりと複数の VLAN 経路の設定を行なって，結果的にクラスタリングに相当する動作を行う．これにより位置に基づくヒューリスティックなオーバレイネットワークルーティングとデータ転送を可能とし，ネットワーク構成用の各基地局からの制御信号のやりとりを近傍基地局との間に抑えつつ，長距離のデータ転送を行うことを可能とする．品質のゆらぎが大きい無線リンクとは別に品質が確かで資源的にも任意に拡張しやすいメモリ資源と基本的に不変な位置情報に依ったネットワ

ーク構成機構の組み込みは NerveNet が担うべき社会的な役割に沿うものであると考える．

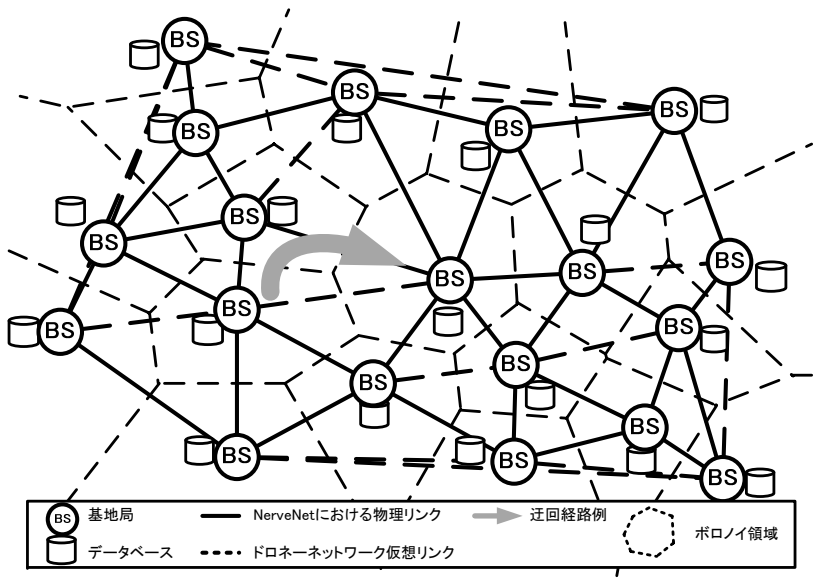

図 3　NerveNet 上のドロネーオーバレイネットワークの構成

図 3 に, NerveNet におけるドロネーオーバレイネットワークの構成について示した．各基地局は近傍で見通しが効く幾つかのノードと物理的なリンクを結び，全体として連結グラフとなるネットワークトポロジを形成する．ドロネーオーバレイネットワークのトポロジは相対近傍グラフの一種であり, その定義と照らしあわせても, NerveNet が持つ物理リンクと同様のリンクを持つ可能性は高い．しかし，ドロネーオーバレイネットワークにおけるコネクションが NerveNet における隣接ノードとは完全に一致していることはほぼ無い. もちろん NerveNet の L2 レイヤにおいて NerveNet における全ノードと通信できる場合には，ドロネーオーバレイネットワーク層でのコネクションを確立することについて問題は起こらない．しかし，L2 レイヤの VLAN 設定以前のどこまでの基地局と VLAN を接続するのかを判断する状況においては，L2 レイヤにおける到達性の保証もない．その様な場合を想定し，ドロネーオーバレイネットワークに迂回経路の概念を導入する．物理リンクが，あるノード間においてのみ通信可能であるという仮定を置き，ドロネーオーバレイネットワークのレイヤでの迂回ルーティングを行う．通常のドロネーオーバレイネットワークの自律分散構成アルゴリズムに迂回ルーティングテーブルの構成手法を導入し，ルーティングルールの改良も行う．図２にて点線で示した仮想リンクは，他の NerveNet の物理リンクを乗り継ぐことによって達成される．図中の迂回経路例は，ある BS 間の仮想リンクと同等の価値を持つ物理リンクの乗り継ぎ方を示している．この様に迂回経路を含めつつもドロネーオーバレイネットワークを構成した場合，以下のような利点がある．

- 位置に依るグリーディルーティングによる全基地局への到達性が保証可能.
- ドロネー図における隣人が一意に確定されデータの冗長管理など分散データベースにおける機構を設計する際に利用が可能.
- ドロネー図の双対グラフであるノード勢力範囲図であるボロノイ図を隣人基地局情報のみで計算することができ適切な空間の分割管理が可能.

以上の利点は，NerveNet の地域の無線アクセス網という役割と非常時の動作を保証する目的を達成するうえで重要である．

2.2　迂回経路を用いたドロネーオーバレイネットワーク構成法の概要

ドロネー図構造のネットワークでは，(i) 地理的に近いノードとのみ接続し，(ii) ノードの隣人表が小さく，(iii) ネットワークの拡張性があり，(iiii) 任意のノード間を到達可能である，などの理由からノード位置を利用したネットワークに適用しやすいという特徴がある[4]〜[6]．しかし，従来提案手法では，各ノードが所持する隣人表(以下リンク) はドロネー辺で繋がるリンクとして定義しており，ローレイヤのネットワークに任意のノードへの到達性の保証がなされていない場合に利用することができない．NerveNet においても L2 スイッチの VLAN タグ数などの資源は有限であり到達性が保証されているとは限らない，だが，到達性が保証されていない中でのドロネーネットワークの生成が完全自動構成手法を確立する上で重要である．

そこで，本稿では，各ノードがドロネーリンクとして所持する隣人内で，ノードの通信範囲外に存在する隣人に対してはリンクを仮想的に構成し，その仮想リンク間を迂回経路によって通信可能とすることで，直接的に無線リンクで接続した基地局ノード同士でしか通信できないと仮定した NerveNet の初期環境に適用することを考える．

本研究では，通信範囲のモデルとして Unit Disk Graph(以下，UDG) を用いて定式化を行う．この UDG モデル上において，各基地局ノードはドロネー図生成のための従来アルゴリズムを拡張し，局所ドロネー化及び基地局ノード間へのリンク通知を繰り返し実行することで，提案ネットワークを自律分散的に構成する．特に，リンク通知は，自身の隣人表から特定の 2 ノードを選択し，その 2 ノード間が通信範囲より長い場合においては，2 ノード間の迂回経路の生成を行い，迂回経路とともにノードの情

報を通知することによって，迂回経路による仮想リンクを構成することができる．また，生成した迂回経路をそのまま利用すると，仮想リンク間の迂回経路に冗長な経路が存在する場合がある．そのため，迂回経路を短縮する縮退法についても述べる．さらに，生成ネットワークの利用法として，欲張り法およびその拡張法によって迂回経路を用いて仮想リンクを辿る欲張り法およびその拡張法による経路選択法についても述べる．

3. ドロネーオーバーレイネットワーク経路表の分散生成法

3.1 準備（UDG）

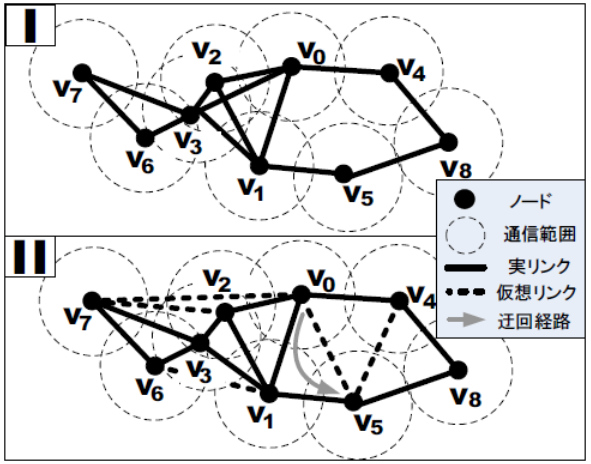

図 4　UDG 上で生成するドロネーネットワーク

ここでは，各基地局ノード（以下ノード）の前提についてまとめる．ノードは以下の前提に基づくものとする．

（１） 各ノードは，すべて同じ処理能力をもち，マルチホップ無線基地局内の計算主体である．

（２） 各ノードは，識別可能な固有の ID 及び，GPS などで自ノード位置(X/Y 座標) 通信半径を所持するものとする．

（３） 各ノードは，通信範囲外のノードに対しての通信は作成中の経路表により構成されるオーバレイネットワークによるマルチホップ転送を行い，直接もしくはマルチホップ転送によって，他ノードと(2) の情報について交換可能である．

このような前提を表すモデルとして Unit Disk Graph(以下 $UDG(V)$) を用いる．$UDG(V)$ とは，全ノードの通信範囲の半径を r として，任意の 2 ノード $vi, vj \in V$ において，$|v_iv_j| < r$ を満たす 2 ノードが接続する無向グラフである(ここで，V はユークリッド平面上のノード集合，$|v_iv_j|$ は v_i,v_j 間の辺 v_iv_j の距離を表す)．$UDG(V)$ はこの定義に基づくため，平面グラフではなく，辺が交差する場合が存在する．また本稿で用いるネットワークは，与えられた任意の 2 ノード間の経路の存在を仮定するため，$UDG(V)$ が連結グラフであるとする．

図 4 の(I) に 9 個のノードから成る $UDG(V)$ を示す．図上では，通信範囲の直径を前述の r とするため，ノードを中心とした円が交差する場合にノード間が繋がる．そのため，図上では v_0 と v_1 間は繋がるため，通信経路が存在する．

Note: また，ここでは説明の簡単化のため，各ノードは同期的に計算処理を実行するが，非同期の場合にも対応できる．

3.2 準備（仮想リンク）

図 4 の(II) に $UDG(V)$ 上から，ノード間をドロネー辺で繋がる，提案ネットワークの例を図示する．図上では v_0 は，通信範囲内のノード$\{v_1,v_2,v_4\}$ とは実線で繋がり，通信範囲外のノード$\{v_5,v_6,v_7\}$ は破線によって繋がり，これらをノードが所持する隣人表とする．このように，ノードが通信範囲内のノードと繋がる辺を実リンク，直接通信可能でないノード繋がる辺を仮想リンクと定義する．また，これらの辺の両方をまとめてリンクと呼ぶ．

以上の関係をまとめると，任意のノード $v_i \in V$ が所持するノード集合を V_i と示し，$vj \in Vi$ が，$r < |v_iv_j|$ を満たす場合にがノード v_i は v_j と仮想リンクで繋がる．

図 4 の(II) において v_0 の所持する仮想リンクで繋がるノード v_5 と通信を行う場合には，実リンクを辿り，マルチホップ通信を行うことで v_5 と通信を行う．そのため，v_0 はノード v_5 と通信を行うために実リンクの経路が必要である．この実リンクの経路を迂回経路と定義する．

ノード $v_i \in V$ から仮想リンクで繋がるノード v_j までの迂回経路を $Path(v_i, v_j)$ によって表す．また，$Path(v_i, v_j)$ 上の具体的な経路を示す場合には，v_i から近いほうから順番に$(v_a, v_b, ...v_n)$と表す．迂回経路は順番も考慮するため，$2 <= |Path(v_i, v_j)|$ の場合には，$Path(v_i, v_j) = | Path(v_j , v_i)$ となる．

Note: ただし，仮想リンクに対する迂回経路は複数存在する．そのため，仮想リンク間の経路を常に最小のホップ数で到達できるとは限らない．図 4 の(II) では，v_0 の v_5 までの迂回経路 $Path(v_0, v_2)$ は，(v_1) 以外にも(v_4, v_8) などが考えられる．

Vol.2010-MBL-55 No.17
2010/9/3

3.3 全体的な生成手順

全体的なドロネー図の自律分散生成法について述べる．*UDG*(*V*) 上のノード $v_i \in V$ が実行するドロネー図生成築アルゴリズムは所持する隣人数$|V_i|$ によって異なり，

($|V_i| = 1$) の場合では，v_i は隣人表に 2 つ以上のノードを所持するまで他者からの通知を待つ．

($|V_i| = 2$) の場合では，v_i は三角化通知のみを行う．

($|V_i| >= 3$) の場合では，v_i は局所ドロネー化，委譲，三角化通知の 3 つの処理を順番に行う．

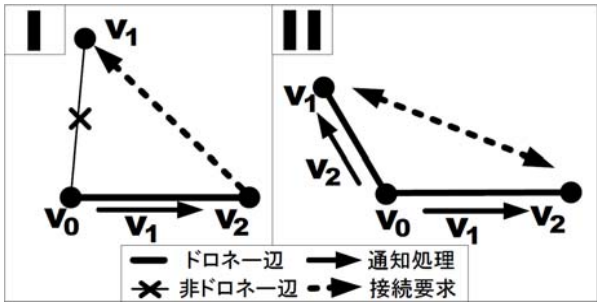

図 5　ドロネー図生成時の通知処理と接続要求

なお，初期状態において v_i は，$|v_i v_j| < r$ となる全ての v_j を V_i のノードとし，他ノードとの接続要求や通知を行うことによって V_i を更新する．これを全てのノード

が繰り返し実行することで，全体としてドロネー図を構築する．ここで，上記で示した主要 3 つのアルゴリズムの内容について確認する．

局所ドロネー化とは，v_i を中心として時計順に並べ全ての隣人ノードを，外接円判定によって v_i とドロネーリンクで繋がるノード（ドロネーノード）か，繋がらないノード（非ドロネーノード）かの 2 種類に分類する処理である．

委譲とは，v_i の局所ドロネー化によって非ドロネーノードをドロネーノード内でその非ドロネーノードに最も近いノードへ委譲する通知処理である．また，三角化通知とは，ドロネーノード間が互いノード間の関係を認識させるために行う通知処理である．図 5(I) に委譲処理を示す．v_0 はドロネーノードである v_2 に非ドロネーノードである $v1$ を通知する．ここで，v_0 から通知を受け取った v_2 は v_1 に対して通信経路の接続要求を行うが v_2 の通信範囲内にないため，直接接続できない．同様に，(II) の三角化通知においても，v_0 は v_1,v_2 の 2 つのドロネーノード間が相互に接続するため，両方に対して通知を行うが，通知を受けたノードが自身の通信範囲内に存在しない場合には接続できない．

そこで，ノード $v_i \in V$ がこれらの通知処理を行う場合には，迂回経路を付加して通知することでこれに対応する．これらの通知処理は，v_i が所持するノードから対象となる 2 つを取り出し，そのノード間の関係に基づき通知処理を行う．そのため，次節では，2 つのノード間に対する迂回経路の生成及びその縮退法について説明する．

Note: なお，ここではノードの隣人による迂回経路の生成は先に来た通知を優先する．つまり，v_i が v_j からの通知処理によって $Path(v_i, v_k)$ の通知を受けた場合に，v_j 以外の他のノード v_l から v_k 及び $Path(v_i, v_k)$ の通知を受けると，v_l からの通知は破棄する．

3.4 迂回経路生成

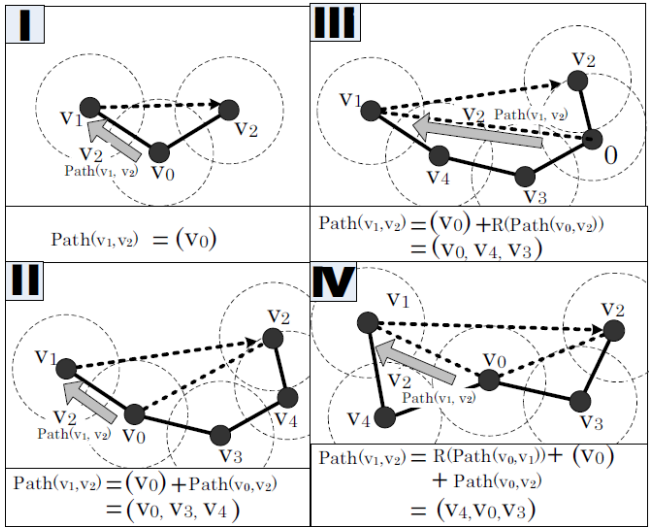

図 6 迂回経路の生成例

本節では，各ノードによる迂回経路の生成アルゴリズムについて Algorithm1 及び図 5 を用いて説明する．ここでは，ノード $v_i \in V$ が隣人表から 2 つのノード v_j , $v_k \in Vi$ を取り出した場合において，v_i が v_j に対して v_k の情報を通知する時における迂回経路 $Path(v_j , v_k)$ の生成について示す．また，v_j ,v_k 間の迂回経路を $Path(v_j , v_k) = null$ としてアルゴリズムを始める．これらは Algorithm 1 の 2 行目に初期状態として与えている．

Step1. v_i は $r < |v_j v_k|$ であるかをチェックする．これは Algorithm 1 の 2 行目に当たる．
Step2-1. v_i は v_k 及び v_j の両方と実リンクで接続される場合，v_i 自身のみが v_j と v_k 間の迂回経路となるため，$Path(v_j, v_k)$ に vi 自身のみを追加する．これは Algorithm 1 の 3,4 行目に当たる．また，図 3(I) のように v_0 の所持するノード v_1, v_2 が実リンクで繋がっているとこの処理を実行する．

Algorithm 1 ノード v_i による迂回経路生成

```
1: Given v_j , v_k ∈ V_i, Path(v_i, v_j ) ← null
2: if r < d(v_j , v_k) then
3:    if v_i has no Path then
4:        Path(v_j , v_k) ← v_i
5:    else if v_i has Path(v_i, v_k) then
6:        Path(v_j , v_k) ← (v_i + Path(v_i, v_k))
7:    else if v_i has Path(v_i, v_j ) then
8:        Path(v_j , v_k) ← (Reverse(Path(v_i, v_k)) + v_i)
9:    else
10:       Path(v_j , v_k) ← (Reverse(Path(v_i, v_k)) + v_i + Path(v_i, v_k))
11:   end if
12:end if
```

Step2-2. v_i の所持する v_k が他ノードから通知を受けたため，仮想リンクで繋がる場合には，迂回経路 $Path(v_i, v_k)$ を所持している．そのため，v_i はまず $Path(v_j , v_k)$ に v_i を追加した後，$Path(v_i, v_k)$ を順番に追加する．これは Algorithm1 において 5,6 行目に当たる．なお，アルゴリズム中 7 行目の”+”は Path の連結を示す．つまり，$Path(v_i, v_k) = (v_a, v_b)$ の場合，$Path(v_j , v_k)$ は(v_i, v_a, v_b) となる．図 6(II) において，v_0 は，$Path(v_1, v_2)$ に自身を格納した後，$Path(v_0, v_2)$ を格納する．
Step2-3. v_i の所持する v_j が仮想リンクで繋がる場合には迂回経路 $Path(v_i, v_j)$ を所持しているため，v_i は $Path(v_j , v_k)$に反転させた $Path(v_i, v_j)$ を追加した後，v_i を追加する．Algorithm1 中 7,8 行目に当たり，$Reverse(Path(v_i, v_j))$とは，経路の反転を表す．つまり，$Path(v_i, v_j) = (v_c, v_d)$の場合には $Reverse(Path(v_i, v_j) = (v_d, v_c)$ となり，結果，$Path(v_j , v_k) = (v_d, v_c, v_i)$ となる．v_j から v_i への経路は，v_i から v_j への経路の反転であるため，このような処理を行うことで，v_j から v_k までの辿ることが可能な経路を構成できる．これは，図 6 中(III) に当たる．
Step2-4. vi の所持する v_j 及び v_k の両方が仮想リンクで繋がる場合には迂回経路 $Path(v_i, v_j)$ 及び $Path(v_i, v_k)$ を所持するため，v_i は $Path(v_j , v_k)$ に反転させた $Path(v_i, v_j)$ を追加した後，v_i を追加し，さらに $Path(v_i, v_k)$ を追加する．これは，Algotithm1 中 9,10 行目，図 6 における(IV) に当たる．

3.5 迂回経路の縮退

ノード v_i が v_j と v_k 間の迂回経路 $Path(v_j , v_k)$ の生成時，v_i が所持する $Path(v_i, v_j)$ 間と $Path(v_i, v_k)$ 間に共通するノードが存在する場合がある．この場合，$Path(v_j , v_k)$ を生成して v_j にそのまま通知を行うと，$Path(v_j , v_k)$ は v_j と v_k 間の通知処理時に共通するノードを 2 回経由する冗長な通信経路となるため，ネットワーク全体の通信量の増大につながる．ここでは，このような冗長な経路に対して 2 種類の縮退アルゴリズムについて述べる．

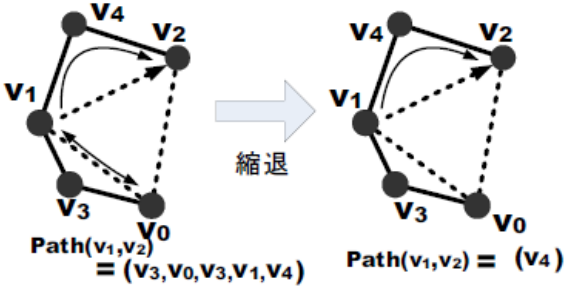

図 7 縮退法 1 による縮退例

Algorithm 2 ノード v_i による縮退法 1

```
1: Given Path(v_j , v_k), n = |Path(v_j , v_k)|, v_j , v_k ∈ V_i
2: for x = 1 to n do
3:     if v_x = v_j then
4:         for a = 1 to x do
5:             Delete(v_a)
6:         end for
7:     else if v_x = v_k then
8:         for b = x to n do
9:             Delete(v_b)
10:        end for
11:        x ← n
12:    end if
13: end for
```

[縮退法 1]

ノード v_i がノード v_j から v_k 間の迂回経路 $Path(v_j, v_k)$ を v_i 以外の他ノードの通知によって得た情報によって構成すると，$Path(v_j, v_k)$ 内に v_j または v_k が含む場合が存在する．ここで v_i の通知によって v_j を含む $Path(v_j, v_k)$ を v_j が取得すると,v_j が v_k に通信を行う場合に，通信パケットは v_j から v_j までの経路を辿った後，v_j から v_k までの経路を辿る．また，同様に v_k が v_j からの通信パケットを受信する時，通信パケットは v_k までの経路を経由して v_k に到達する．そのため，$Path(v_j, v_k)$ 内に v_j が含む場合は v_j までの経路及び v_k 以後の経路を削除できる．以下に，縮退アルゴリズムについて Algorithm 2 を用いて述べる．なお，迂回経路長$|Path(v_i, v_j)|$ を n とし，$Path(v_j, v_k)$ は v_j から近いノードから順番に，迂回経路リストの番号を$(0 <= x <= n)$ とする．

Step1. まず vi は，所持する $Path(v_i, v_j)$ のうちノード v_x(初期値は $x = 1$，つまり v_1) を取り出し,$v_x = v_i$ または $v_x = v_k$かを検査する．これは Alogorithm 2 中 3,7 行目にあたる．

Step2-1. v_x が v_i でも v_k でもないと $x \leftarrow x+1$ として Step1 に戻る．これは Algorithm 2 中 2 行目にあたる．なお，$n = x$ の場合には処理を終了する．

Step2-2. $v_x = v_i$ の場合には，v_1 から v_x 間の全てのノードを削除し，$x \leftarrow x + 1$ として，Step1 に戻る．これは Algorithm 2 中 3 から 6 行目に当たり，a の値を 1 から x まで変化させ，その間のノードをすべて削除する．なお，$x = n$ の場合には処理を終了する．

Step2-3. $v_x = v_k$ の場合には，v_x から v_n までのノードを全て削除し，処理を終了する．これは Alogorithm 2 中 7 から 11 行目に当たる．これを図 7 を用いて説明する．図上では v_0 が生成する $Path(v_1, v_2)$ が$(v_3, v_0, v_3, v_1, v_4)$ であり，これを上記のアルゴリズムによって縮退する．まず，$Path(v_1, v_2)$ から v_x を順番に検査すると，4 番目に v_1 が存在する．そのため，Step2-2 によって v_1 を含めた v_1 以前のすべてのノードを削除し，$v_x = v_4$ として，v_4 の検査を終えると処理を終了する．この縮退法により $Path(v_1, v_2)$ は (v_4) のみとなる．

[縮退法 2]

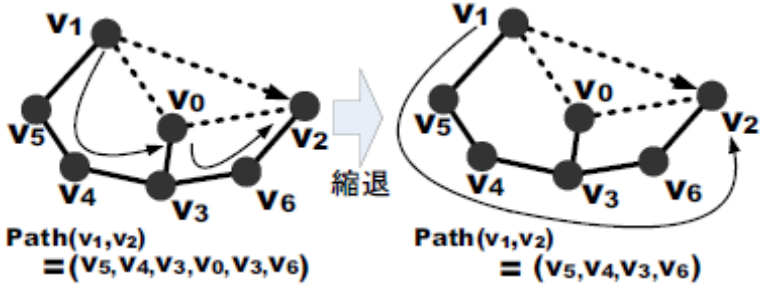

図 8 縮退法 2 による縮退例

さらに，$Path(v_j, v_k)$ 内に v_j ,v_k を所持せず，迂回経路に共通するノードが存在する場合がある．そこでこのような場合における短縮アルゴリズムについて Algorithm 3 を用いて示す．また，ここでは前述と同様，$n = |Path(v_i, v_j)|$ とし，v_j から近い順番として，v_x, v_y の x, y を迂回経路のリスト番号とする$(0 <= x, y <= n)$.

Algorithm 3 ノード v_i による縮退法 2

```
1: Given Path(vi, vj ), n = |Path(vi, vj )|
2: for x = 1; to n do
3:     for y = x + 1 to n do
4:         if vx = vy then
5:             while vx = vy+1 do
6:                 Delete(vy)
7:                 y ← y + 1
8:             end while
9:         end if
10:    end for
11: end for
```

Step1. まず v_i は，所持する $Path(v_i, v_j)$ のうちノード v_x(初期値は $x = 1$，つまり v_1) を取り出す．次に，$y \leftarrow x+1$ とした v_y を取り出し，$v_x = v_y$ かを検査する．これは Algorithm 3 中 4 行目に当たる．

Step2-1. $v_x = v_y$ でない場合には，$y \leftarrow y + 1$ として Step1 に戻る．これは Algorithm 3 中 3 行目に当たる．

Step2-2. $v_x = v_y$ の場合には，v_x から v_y-1 間の全てのノードを削除し，$x \leftarrow y, y \leftarrow y + 1$ として，Step1 に戻る．これは Algorithm 3 中 4 から 9 行目であり，v_yから v_x 間の番号のノードを削除している．

Step3. $y = n$ となると，$x \leftarrow x + 1, y \leftarrow x + 1$ として Step1 に戻る．これを $x = n$ となるまで繰り返し，$x = n$ となる時点で処理を終了する．図 8 を用いて説明する．$v0$ が生成する迂回経路 $Path(v_1, v_2)$が$(v_5, v_4, v_3, v_0, v_3, v_6)$ である場合において上記の Algorithm 3 によって縮退する．まず，v_x を v_5, v_4 として同じノードがないかを検査するここで，$v_x = v_3$ の検査中，$v_y = v_3$ となるため，v_x から v_y-1 までのノードを迂回経路リストから削除する．ここでは，v_x から v_y-1 の要素が v_3, v_0 であるため，これらを削除する．そして $v_x = v_3, v_y = v_6$ として検査を再開し，$v_x = v_6$ となった時点で縮退

アルゴリズムを終了する．v_i はこれらの迂回経路の生成及び縮退アルゴリズムを終了すると，三角化通知または委譲処理によって v_j に v_k と迂回経路 $Path(v_j, v_k)$ の二つの情報の通知を行う．これによって，$|v_jv_k| > r$ の場合においても，v_j は v_k に迂回経路 $Path(v_j, v_k)$を辿ることで v_k と通信を行うことができる．この迂回経路の生成によって，実リンクで繋がらないノード間に対しては，迂回経路を経由させることで仮想リンクを構成できる．そのため，各ノードが従来のアルゴリズムに加えてこれらのアルゴリズムを繰り返し実行することで，迂回経路を用いたドロネーネットワークを自律分散的に生成できる．

4. 経路表の分散生成法のシミュレータの試作

提案手法による経路表生成手法の動作確認の為，Java によってシミュレータを試作した．有理数クラスを自作し，計算誤差が発生しない実装を行った．１００ノードの UDG を生成した上での動作確認を行い，ドロネーネットワークの生成について確認し，アルゴリズムの正常動作を確認した．

5. 関連研究

無線環境を考慮したネットワーク上での経路選択は，スケーラビリティを保持するため，フラッディングの利用を避ける必要がある．そこで，ID やノード位置に基づくネットワークの構造化や対象範囲の制限をした通信によって，ネットワーク全体の通信量を低減しつつ，任意のノード間に対して到達可能な経路選択法が研究されている．

Karp ら[7] は，ノードの位置関係によってガブリエルグラフを構築し，欲張り法及び Right-Hand-Rule を利用することで，フラッディングを利用せずに，平面グラフ上で任意のノード間での経路選択を保障している．一方，提案手法では，構成したドロネーリンクを辿ることによって任意の 2 ノード間で経路選択を行うことができる．Li 及び Wang ら[8], [9] は，通信範囲内のノードに焦点を当て，UDG 上から平面局所化ドロネー図と呼ばれる平面グラフを提案し，同構造を利用した欲張り法による経路選択について議論している．これに対して提案手法では，通信範囲外のノードには迂回経路によって到達可能なリンクを構成できるため，グラフが連結であれば，通信範囲に依存せずネットワークを構築することができる．

Liao [10] らの手法では，平面を格子状に分割し，各格子で代表ノードを割り当て，代表ノードのみが格子間で経路情報を交換し，格子内ノードのみに取得した経路情報をマルチキャストする．これによって，全格子内の任意のノード間で経路選択を可能にしている．しかし，この手法は各格子に割り当てられた代表ノードに負荷が集中する．また，Caesar ら[11] らは，Chord 環を利用しており，各ノードがノード ID の近いノードを隣人表として所持し，ID 空間上で近いノード ID を順番に辿ることで ID のみによる経路選択法を実現している．また，ID 空間上の隣人がノードの通信範囲外にある場合には，マルチホップ通信による隣人までの経路を所持することで対応している．しかしながら，ノード数の増加に伴い ID 空間も大きくなるため，ノード数が大きい場合には隣人までのホップ数が増加する．一方，提案手法では，ノード数の増加に対して所持する隣人数は一定であり，拡張性が高い．

6. おわりに

本研究では，マルチホップ網を軸にした地域サービスプラットフォームである NerveNet 上にドロネーオーバレイネットワークを生成する方法について提案した．実装に向けてドロネーオーバレイネットワーク生成アルゴリズムの直行 3 次元座標上の位置使用への対応や NerveNet のスイッチレイヤとの連携機構について研究を行う．

謝辞 本研究は，総務省 戦略的情報通信研究開発推進制度(SCOPE)の助成を受け行っている．

参考文献

1) M. Inoue, et al., “A Novel Managed Wireless Mesh Architecture for Community Service Platform,”IEEE WCNC, Apr. 2009.
2) 大西ほか, “分散処理･制御アクセスネットワーク NerveNet によるインタラクティブ広告配信,” 信学技報, MoMuC2010-1, 2010 年 1 月
3) 阪神・淡路大震災調査報告編集委員会『阪神・淡路大震災調査報告 ライフライン施設の被害と復旧』（社）土木学会(1997/9)
4) 大西真晶, 源元佑太, 江口隆之, 加藤宏章, 西出亮, 上島紳一 :“ノード位置を用いた P2P モデルのためのドロネー図の自律分散生成アルゴリズム”，情報処理学会論文誌：データベース, **47**,SIG4(TOD29), pp. 51 － 64 (2006).
5) F. Ara´ ujo and L. Rodrigues: “Geopeer: A location-aware peer-to-peer system”, Proceedings of the 3rd IEEE International Symposium on Network Computing and Applications (IEEE NCA04), Cambridge, MA, USA, pp. 39−46 (2004).
6) P. Bose and P. Morin: “ISAAC: 10th international symposium on algorithms and computation”, ISAAC, pp. 113−122
7) B. Karp and H. T. Kung: “GPSR: greedy perimeter stateless routing for wireless networks”, MobiCom ’00: Proceedings of the 6th annual international conference on Mobile computing and networking, pp. 243−254 (2000).
8) X.-Y. Li, G. Calinescu, P.-J. Wan and Y. Wang: “Localized delaunay triangulation with application in ad hoc wireless networks”, IEEE Transactions on Parallel and Distributed Systems, **14**, 10, pp. 1035−1047 (2003).
9) Y. Wang and X.-Y. Li: “Efficient delaunay-based localized routing for wireless sensor networks:

Research articles”, International Journal of Communication Systems, **20**, 7, pp.767−789 (2007).
10) M. Caesar, M. Castro, E. B. Nightingale, G. O’Shea and A. I. T. Rowstron: “Virtual ring routing: network routing inspired by dhts”, SIGCOMM, pp. 351−362 (2006).