フラッディング高速化手法の実装と評価

瀧本栄二[†1] 滝沢泰久[†1]
鈴木龍太郎[†1] 小花貞夫[†1]

アドホックネットワークは，ノード間の自律的な通信により，柔軟なネットワークの構築が可能である．そのため，様々な分野での適用が期待されている．アドホックネットワークにおける通信性能の評価指針の 1 つに，通信遅延がある．しかし，従来のアドホックネットワークに関する研究では，通信遅延の短縮が重視されていない．たとえば，アドホックネットワークの適用アプリケーションである車車間通信等では，低通信遅延が求められる．そこで，本論文では，アドホックネットワークにおける通信方式の 1 つであるフラッディングに関して，低通信遅延を実現する構成手法を提案する．提案手法では，フラッディングに関する内部処理をデバイスドライバで行うことで通信遅延を削減する．提案手法のプロトタイプシステムを用いた評価を行った結果，従来手法に比べ 1 ホップあたりの通信時間増加率を 30%削減することができた．

An Implementation and Evaluation of an Accelerating Method for Flooding

Eiji Takimoto,[†1] Yasuhisa Takizawa,[†1] Ryutaro Suzuki[†1] and Sadao Obana[†1]

Ad-hoc network can construct flexible networks by an autonomous communications among nodes. Therefore, ad-hoc network is expected to apply in the number of different fields. The communication latency is one of important issues in ad-hoc network. In the conventional researches, however, it is not enough to consider the communication latency. For example, the low communication latency is required in the inter-vehicle communication which is one of ad-hoc network applications. Therefore, in this paper, we propose the construction method for reducing the communication latency in the flooding which is a communication method in ad-hoc network. The proposed method can decrease the communication latency by executing the internal processing for flooding within a device driver. Experimental results of the prototype system show that the proposed method can decrease a rate of increasing communication time per hop about 30%.

1. はじめに

アドホックネットワークは，ネットワークを構成するすべての通信ノードが送信，受信，中継を行う機能を持つため，柔軟かつ動的にネットワークを構築することが可能である．アドホックネットワークで行われる通信の 1 つに，フラッディングがある．フラッディングは，アドホックネットワークにおけるブロードキャスト通信であり，ネットワーク全体にパケットを送信するために用いられる．フラッディングでは，あるノードがブロードキャストパケットを送信すると，それを受信したノードがさらに中継を行うことで，ネットワーク全体にパケットが到達する．フラッディングは経路構築時に必要となる端末の探索に使用されることが多いが，パケット到達率に優れるために[1]，マルチキャストの 1 手法として使用される等データ通信にも用いられる[2]．

アドホックネットワークを用いたアプリケーションの 1 つに，ITS（高度交通情報システム）の車車間通信[3] がある．車車間通信では，安全運転支援を目的とした車両情報の配布と収集が行われる．また，対向車の存在や急制動の発生を知らせるための危険情報通知も行われる．これらの通信は，特定の車両間で行われるユニキャスト通信ではなく，周辺一帯に存在する車両全体を対象としたブロードキャスト通信，すなわ

†1 株式会社国際電気通信基礎技術研究所適応コミュニケーション研究所
Advance Telecommunications Research Institute International, Adaptive Communications Research Laboratories

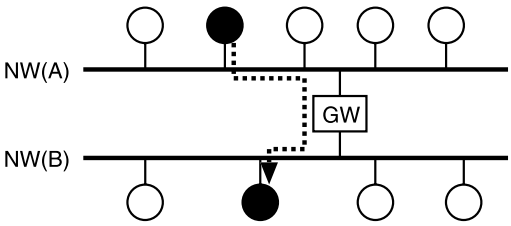

図 1 ネットワーク間接続のための中継
Fig. 1 Relay for inter-network connection.

ちフラッディングである．また，危険情報を速やかに通知する必要があるため，ネットワーク全体にパケットを到達させるまでにかかる通信遅延の削減が重要となる．

アドホックネットワークで行われるマルチホップ通信では，複数のノードによってパケットが中継されるため，送信ノードから離れるほど中継回数が増え，結果として遅延が増加する．したがって，通信遅延を削減するための 1 手段として，中継による遅延（以下，中継遅延と記す）の削減が有効であると考えられる．ここで，中継遅延とは，パケットの受信からパケット中継のために必要となるヘッダの確認と書き換え等を行い，再度送信するまでにかかる遅延を指すものとする．

中継遅延を削減する最適な手段は，中継に関わる処理を 3 層スイッチのようにすべてハードウェア化することである[4),5)]．しかし，アドホックネットワークにおける中継機能のハードウェア化には，解決すべき問題が多く残されている．そこで，本論文では，移動性のある環境下において既存の IEEE802.11 無線通信デバイスを使用した同一チャネルによるアドホックネットワークにおいて，高速な中継を実現する中継および送受信機構の構成手法を提案する．従来のフラッディング中継機構は，**図 1** に示すように IP 通信等で行われるネットワーク間を接続するための中継処理をそのままアドホックネットワークに適用している．しかし，アドホックネットワークにおける中継処理は，**図 2** に示すように，同一ネットワーク内において直接電波が届かないノード間を補完するものである．提案手法はこの違いに着目し，従来の中継機構ではデバイスドライバよりも上位層で実現されていた中継処理をデバイスドライバ内に構成する．また，通信遅延をより削減するために，フラッディングを利用するアプリケーションが直接デバイスドライバ内のフラッディング処理機能を利用するための機構をデバイスドライバ内に実現する．提案手法は，デバイスドライバ内部にフラッディングの送受信と中継機構を構成するため，プロトコルスタックによる処理を削減し，かつその通信の特性を利用して割込み処理時に行われる遅延処理により発生する遅延時間を削減することを可能にする．本論文における中継処理とは，特に明記しない限り図 1 に示すネットワーク間接続のための中継処理ではなく，図 2 に示すアドホックネットワークにおいてノード間を補完するための同一ネットワーク内における中継処理を指すものとする．

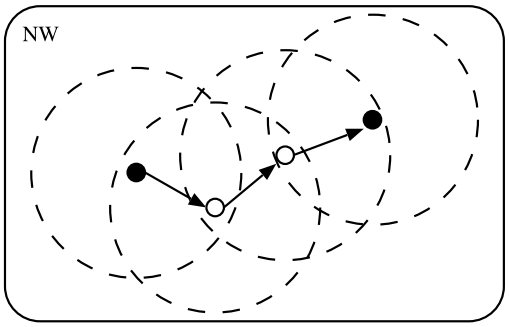

図 2 アドホックネットワークにおける中継
Fig. 2 Relay of adhoc networks.

以下，本論文では，2 章で従来のフラッディング実装手法について述べ，3 章で提案手法とそのプロトタイプ実装について述べ，4 章で提案手法の性能評価を行った結果とその考察を述べる．また，5 章で関連研究について述べたあと，6 章で本論文をまとめる．

2. 従来手法の概要

2.1 ネットワーク間接続のための中継処理

有線ネットワークでは，ルータやゲートウェイ等の限られたノードのみが中継処理を行う．有線ネットワークにおける中継処理は，異なるネットワーク間を接続するためのものである．したがって，中継ノードは，図 1 に示すように，中継が行われる両方のネットワークに接続されていなければならない．そのため，中継ノードは，中継の対象となるネットワークごとに通信デバイスが必要である．

図 1 では，ネットワーク A に属するノードがネットワーク B に属するノードに対してパケットを送信している．このとき，2 つのノードが属するネットワークが異なるため，中継ノードがパケットを中継することで通信を実現している．この例における中継ノードの構成は，**図 3** に示すとおりとなる．

2.2 アドホックネットワークにおける中継処理

アドホックネットワークでは，すべてのノードが中継機能を持つ．ただし，アドホックネットワークにおける中継は，図 2 に示したとおり同一のネットワーク

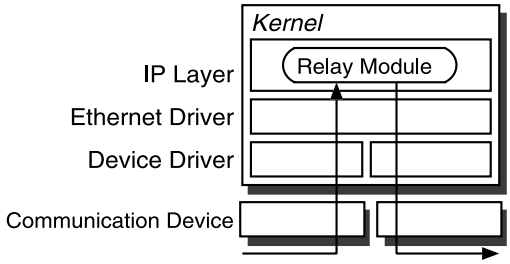

図 3 ネットワーク間接続を行う中継ノードの構成
Fig. 3 Structure of relay nodes for inter-network connection.

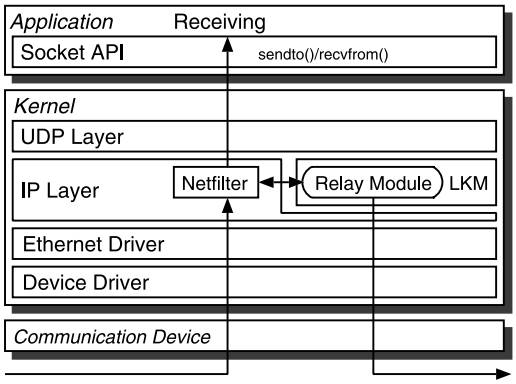

図 4 カーネル内実装手法
Fig. 4 Implementation method into a kernel.

内で閉じて行われる．したがって，各ノードが持つ通信デバイスは 1 つとなる．一般的なアドホックネットワークでは，ネットワーク間接続のための中継処理と同様，IP による通信を行う．アドホックネットワークにおける既存のフラッディング中継機構は，OS のカーネル内で行う実装手法と，アプリケーションによって行う実装手法のいずれかで実装されている．

カーネル内でフラッディングを実現している代表的な実装として，Kernel-AODV [6] がある．Kernel-AODV では，経路要求パケットを UDP でフラッディングしている．パケットの中継は，IP 層で行われる．中継処理を行うためにプロトコルスタックを変更することは，開発コストを上げ，汎用性を低下させる原因になる．そのため，Kernel-AODV では，中継機能を LKM（Loadable Kernel Module）として実装し，Netfilter [7] を用いてプロトコルスタックとは独立した処理を実現している．この実装手法では，受信したパケットは IP 層で Netfilter によってフックされ，LKM に渡される．中継が行われる場合は，LKM 内で中継処理が行われ，IP 層を経由して送信処理が行われる．同時に，受信パケットは，上位の UDP 層へ渡され最終的にアプリケーションが受信する（**図 4** 参照）．

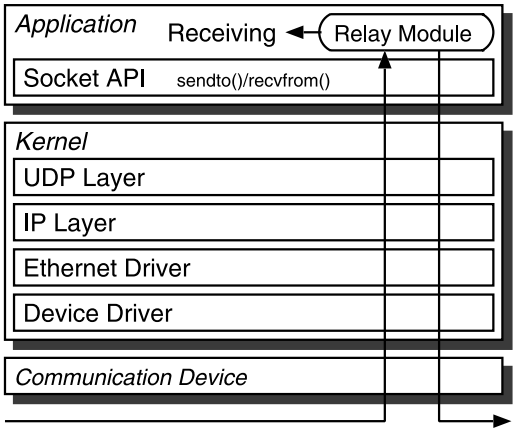

図 5 アプリケーション内実装手法
Fig. 5 Implementation method into an application.

ほかにもアドホックネットワークを応用したメッシュネットワーク[8] では，MCL（Mesh Connectivity Layer）と呼ばれる層を IP 層とデバイスドライバの間に設け，MCL でパケット中継機能を提供する実装手法[9] がある．この手法は，上位のネットワーク層が IP 層に限らず利用することが可能であるが，その本質的な実装手法は，IP 層中なのかその下位なのかが異なる以外は Kernel-AODV と同じである．

より簡易な手法は，アプリケーションで実装することである．この手法では，デーモンのようにフラッディング専用アプリケーションとして実装する方法と，パケットを送受信するアプリケーション自身がフラッディングの制御を行う手法が考えられる[10]．後者は，送受信と中継をアプリケーション内で閉じて行うことができるため，より実装が容易である．この手法による構成と処理は，**図 5** に示すとおりである．

このように，アドホックネットワークにおける中継処理は，ネットワーク間接続のための中継処理と違う特徴を持ちながら，同様またはより通信遅延が大きくなる手法で実現されている．これらの既存技術は，それらが持つ目的を果たすために十分な性能を持つが，提案手法が目的とするフラッディングにおける中継遅延の削減という点では中継遅延が大きいため有効な技術であるといえない．提案手法では，アドホックネットワークの特徴を生かし，低い階層で実現することで通信遅延の削減を実現している．

3. 提案手法とプロトタイプの実装

本論文では，従来カーネル内部もしくはアプリケーション層において実現されていたフラッディングの中継処理を，アドホックネットワークにおける中継処理

特性に基づき，中継におけるすべての処理をデバイスドライバにおいて実現し，フラッディングの高速中継を可能にする構成手法を提案する．さらに，提案手法では，中継時に行われる送受信処理をアプリケーションから利用できる機構を設けることで，より高速なフラッディングを実現する．図 1 で示したように，従来の中継処理は以下にあげる特徴を持ち，その構成は図 3 のとおりとなる．

- ネットワーク間接続を行うことを目的とする．
- 一部のノードのみが中継機能を持つ．
- 接続すべきネットワークの数だけ通信デバイスを必要とする．

ネットワーク間接続のために複数の通信デバイスを用いて中継処理を行う場合，受信時と送信時とで異なる通信デバイスを用いる．中継時には，受信パケットの宛先等に基づいて送信パケットを作成する．また，このときにルーティング処理が行われる．ルーティング処理時には，パケットを送信するための通信デバイスの決定も行われる．その後，送信パケットは，ルーティング処理によって決められた通信デバイスを用いて送信される．このように，ネットワーク間接続では，中継時に複数の通信デバイスを参照することから，中継モジュールがデバイスドライバから独立している必要がある．同様に，ルーティング処理を行うため，IP 層から分離することが困難である．

これに対して，アドホックネットワークにおける中継は，以下の特徴を持つ．

- 同一ネットワーク内で閉じた中継である．
- すべてのノードが中継機能を持つ．
- 必要な通信デバイスは 1 つである．

従来のアドホックネットワークにおけるフラッディングでは，ネットワーク間接続と異なる特徴を持ちながら，デバイスドライバよりも上位層またはアプリケーション層において同様の中継処理を行っている．提案手法では，これらの中継処理の違いとフラッディングの特徴に着目している．フラッディングはブロードキャスト通信の 1 つであるため，つねに宛先に指定するアドレスとしてブロードキャストアドレスを用いる．また，同一ネットワークに対する中継を行うため，送受信に同じ通信デバイスを用いる．したがって，アドホックネットワークにおけるフラッディングでは，ルーティング処理による宛先ネットワークにおけるアドレス検索と，使用デバイスの検索を行う必要がない．

以上から，本論文では，フラッディングの高速化を実現するために，以下の 2 点を基本戦略としたパケット中継機構の構成手法を提案する．

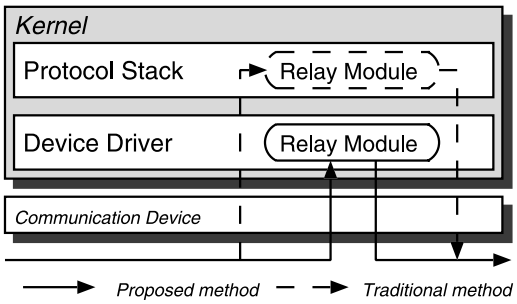

図 6 提案手法と従来手法の違い
Fig. 6 Difference between a proposed method and traditional methods.

- 中継処理をデバイスドライバ内で行う．
- アプリケーションの送受信を直接デバイスドライバが処理する．

すなわち，提案手法では，前者の戦略によりフラッディングにおける中継処理に関わるすべての処理を，デバイスドライバ内で受信割込み処理時に行う（**図 6** 参照）．また，後者の戦略により，送受信時に行われるプロトコルスタック処理を省略し，中継だけでなく送受信の高速化も図る．プロトコルスタックを用いないため，どのノードが送信したかという情報等が欠落する．そこで，提案手法では，フラッディングを行うために最低限必要となる簡素なフラッディング用ヘッダを用いて，フラッディングの制御を実現する．

以下，本章では，提案手法とそのプロトタイプシステムの実装について述べる．

3.1 提案手法の詳細

提案手法では，アドホックネットワークにおける中継処理特性に基づき，中継に関するすべての処理をデバイスドライバ内で行う．デバイスドライバでの中継処理は，パケットの受信に対して即座に対応するため，受信割込みの割込み処理時に行う．これにより，他のカーネル内の処理等による待ち時間を削減する．通信は，フラッディングパケットを受け取るべきノードがすべて受信を完了して終了する．ノード上のアプリケーションが受信するまでに発生する遅延も削減することで，より通信遅延を削減することが可能になる．そこで，提案手法では，IP 等のプロトコルを使用しないことを利用し，送受信時においてアプリケーションが直接デバイスドライバに対してパケットの送受信要求を出すための機構をデバイスドライバ内に実現する．提案手法を用いることで，以下にあげる中継遅延の要因を削減することが可能となる．

- プロトコルスタックの処理
- 遅延をともなう割込み処理

- コンテキストスイッチ（アプリケーション内実装のみ）

3.1.1 プロトコルスタック処理の削減

アドホックネットワークでは，一般的に UDP/IP を用いてフラッディングが行われる．そのため，アプリケーションによる実装手法を除いた従来手法では，IP 層においてパケットの中継処理を行っている．IP を用いた通信では，中継時に以下の処理が行われる．

- 中継すべきパケットかどうかの判断
- 中継先ネットワークの検索（ルーティング処理）
- 中継に使用する通信デバイスの検索
- パケットヘッダの書き換え

フラッディングのようなブロードキャスト通信では，宛先 IP アドレスと IEEE802.11 フレームの宛先 MAC アドレスの両方にブロードキャストアドレスを用いる．そのため，ルーティング処理のように複雑な処理を行う必要がない．これらのことから，提案手法では，最低限の通信プロトコルにより通信を実現する．すなわち，従来の UDP/IP を用いずに簡易なフラッディング用ヘッダを用いて通信を行う．これにより，UDP/IP プロトコルスタックを削減し，中継処理を高速化する．

3.1.2 遅延処理の削減

OS では，カーネルが行う処理を効率化するために，通信データの処理を通信イベントから遅延して行う（以降，この遅延を挟むイベント処理を遅延処理と記す）．遅延処理は，要求された処理を即座に行わずカーネル内のキューにためておき，システムコールやタイマ割込みをきっかけに，キュー内のタスクをまとめて処理する手法である．遅延処理自体は，様々な OS において一般的に用いられている手法である．また，遅延処理は，通信イベントだけでなく，プロトコルスタックの階層間でのパケットの受渡しにも用いられる．無線通信デバイスのデバイスドライバでは，以下の手順でパケットの受信処理を 2 段階に分けて行う．

手順 1 割込み原因を判別し，受信割込みであれば通信デバイスの受信バッファからデータを読み出し，sk_buff 構造体と呼ばれるパケット管理用構造体に記録する．ここで，遅延処理用機構である tasklet に登録されている受信処理用エントリをアクティブにして割込み処理を終了する．

手順 2 システムコールやタイマ割込みを契機に tasklet により処理を再開する．ここでは，受信パケットの種別（ARP，IP，ICMP 等）の判別やエラーチェック等を行い，上位層へ処理を引き渡す．

注目すべきは，手順 1 はパケット受信時に即座に行われるのに対し，手順 2 はカーネルからの呼び出しに対して行われる点である．中継遅延を削減するためには，パケットの受信を駆動契機とする必要がある．したがって，手順 2 の一部として提案手法を実装しても，パケットの受信に同期して処理を行わないため，その効果を期待することができない．提案手法では，このことを考慮し，パケット受信割込み発生時において中継処理を行うことで，中継遅延を削減している．ただし，パケット受信割込み時に処理を行うと，割込み禁止時間が長くなりシステム全体の性能に影響を及ぼす可能性がある．この点を考慮し，提案手法のプロトタイプシステムでは，最も処理オーバヘッドが小さくパケット到達率の高いピュアフラッディングをフラッディングプロトコルに採用している．

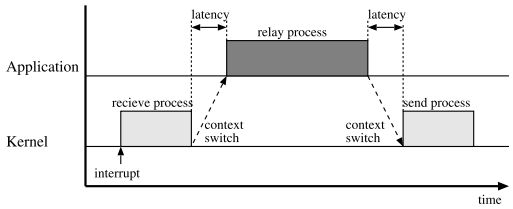

図 7 中継時のコンテキストスイッチ
Fig. 7 Context switch of relay processing.

3.1.3 コンテキストスイッチの削減

アプリケーションによるフラッディングでは，中継処理をアプリケーション自身が行う．中継時には，**図 7** に示すように，カーネルとアプリケーションの間でコンテキストスイッチが発生する．また，アプリケーションでパケットを扱うためには，アプリケーションとカーネルの間でパケットをコピーする必要がある．したがって，アプリケーションで実装する場合，カーネル内で実装した場合と比べ，コンテキストスイッチとメモリコピーによる遅延が増えることになる．また，カーネル内実装とは異なり，中継パケットが UDP 層を往復することも遅延となると考えられる．この中継遅延は，アプリケーション層よりも低い階層において実現することで，削減することが可能である．

3.2 プロトタイプシステムの概要

本論文では，プロトタイプシステムを以下のように構成した．

- 計算機：PC/AT 互換機（CPU: PentiumD 3 GHz）
- 通信デバイス：Prism2.5 チップ使用 IEEE802.11b 無線 LAN カード
- OS：CentOS Linux（kernel2.6.9）
- デバイスドライバ：linux-wlan-ng0.2.3

Linux は，オープンソースであることと，サーバか

ら組み込み用途まで多様な使用実績があることから採用した．通信デバイスには，一般的であり入手も容易であることからIEEE802.11bデバイスを採用した．デバイスドライバは，LinuxのPrism2.5チップ用ドライバであるlinux-wlan-ngを用いた．これらは，既存研究[6),10)]で使用されているシステム構成と同等のものであることから，妥当であるといえる．フラッディングアルゴリズムは，5章で述べているように目的に応じて最適なアルゴリズムが異なる．提案手法では，車車間通信のようにモビリティのある環境での通信遅延の短縮を目標としているため，処理オーバヘッドが小さいこととモビリティ耐性のあることをアルゴリズム選択における条件とした．タイマを用いた手法では，タイマによる遅延が大きくなり，中継遅延の削減を行うことが困難である．確率を用いた手法は，ノード密度等を考慮して適切に中継確率を調整しなければパケット到達率が悪化する．位置情報を用いた手法では，定期的に位置情報をHelloパケットにより得る必要があり，デバイスドライバ内部で実現するには処理が複雑すぎると考えられる．近隣ノード情報を用いた手法も同様であるが，さらにモビリティのある環境では，指定された中継ノードが電波到達範囲から離脱している可能があり，適用が難しいと考えられる．以上から，提案手法では，移動性のある環境でも高いパケット到達率を実現することができ，処理オーバヘッドもその単純さから最も小さいと考えられるピュアフラッディングを採用した．

3.2.1 送受信処理

提案手法では，アプリケーションがデバイスドライバに対して直接送受信を要求する．プロトタイプでは，実装を簡素化するため，ioctlシステムコールを改良した．具体的には，ioctlシステムコールで扱うことができる要求に，フラッディングパケットの送信と受信を加えた．これにより，前章で述べたMACヘッダ変換による遅延を削減した．さらに，ソケットAPIを使用することなく通信を行うことで，より高速化を実現した（**図8**参照）．

フラッディングを行う際には，すでに中継済みのパケットを再度中継することによるループを回避する必要がある．そこで，パケットの冗長性を判断するために，**図9**のフラッディング用ヘッダを用いた．このヘッダは，送信時にデバイスドライバによって作成される．プロトタイプでは，ユーザが送受信するデータにもヘッダが含まれるものとした．ヘッダの詳細は，以下に示すとおりである．

ID ノード識別子である．ID長が変わることによる性能への影響は非常に小さいと考えられるため，プロトタイプではデバイスに割り振られたIPアドレスの下位8ビットを用いた．

SEQ パケットに与えられる番号であり，ノードが新たにフラッディングパケットを生成するたびに1つ値が増加する．パケットの中継処理において，既知のパケットか否かを判断するために用いる．

TTL パケットの生存時間を表し，1回中継されるごとに値が1つずつ減算される．アプリケーションがフラッディングの伝達範囲を指定する際に使われる．たとえば，Helloパケットは，TTLを1に設定することで実現することができる．

LEN パケットのデータ長を表す．ただし，ヘッダ長は含まれない．

DATA データ本体である．2,307バイトは，無線通信で送ることができる最大値である．

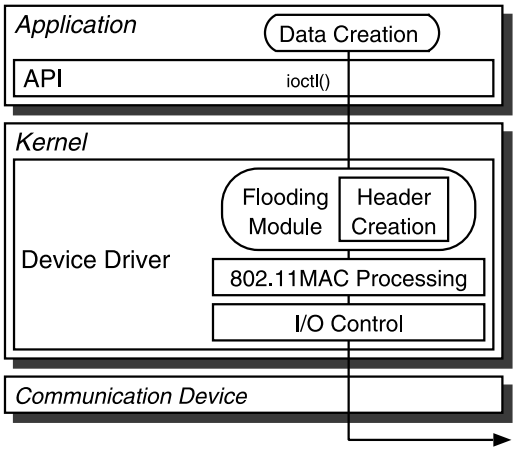

図8 プロトタイプの構成（送信）
Fig. 8 Structure of a prototype (transmit).

ID (8)	SEQ (8)	TTL (8)	LEN (16)	DATA (0-2307Bytes)

ID(8bits):Host Identify Number
SEQ(8bits):Sequential Number
TTL(8bits):Time To Live
LEN(16bits):Data Length(Bytes)

図9 フラッディング用ヘッダ
Fig. 9 Header for flooding.

3.2.2 中継処理

ピュアフラッディングでは，冗長パケットを中継しないように，受信パケットをキャッシュする．提案手法では，中継をデバイスドライバが行うため，デバイスドライバ内にキャッシュを作成した（**図10**参照）．ただし，プロトタイプシステムでは，キャッシュのクリアは行っていない．

プロトタイプでは，ioctlシステムコールによって

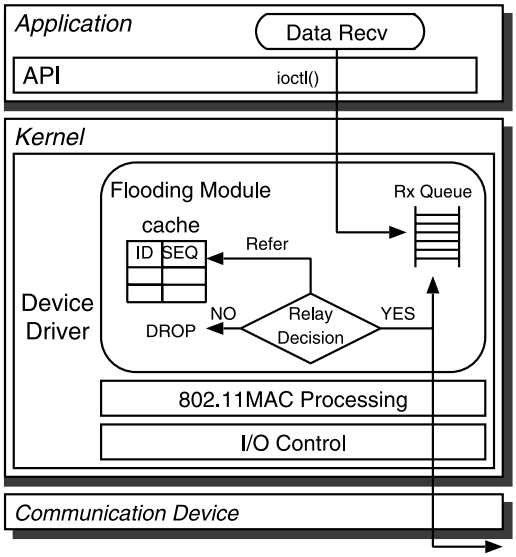

図 10 プロトタイプの構成（受信と中継）
Fig. 10 Structure of a prototype (recieve and relay).

受信が行われる．そのため，キャッシュと同様に，受信用のキューをデバイスドライバ内に追加した．中継処理の手順は，以下に示すとおりである．

手順 1 フラッディングパケットか否かの判断
- フラッディングパケットであれば，以下の手順を行う．そうでなければ，通常の処理（上位レイヤへパケットを渡す）を行う．

手順 2 冗長性のチェック
- はじめて受け取るパケットであれば手順 3 を行い，キャッシュを更新する．
- 冗長パケットであればパケットを破棄して終了する．

手順 3 MAC ヘッダとフラッディング用ヘッダの書き換え

手順 4 受信キューへの挿入と送信処理

手順 1 は，パケット種別の判断である．フラッディング以外の通信を妨害しないためには，受信したパケットがフラッディングパケットであるか否かを判断する必要がある．フラッディングでは，MAC アドレスにブロードキャストアドレスを用いる．しかし，ブロードキャストアドレスは，ARP 要求等様々な通信で用いられる．そのため，プロトタイプでは，MAC ヘッダのタイプフィールドに，他のプロトコルと重複しない数値を用いており，これらの 2 点からフラッディングパケットか否かの判断を行っている．

手順 2 では，パケットの冗長性チェックを行う．デバイスドライバ内には，過去に受信したパケットの ID と SEQ の組をキャッシュしている．ここでは，キャッシュと受信パケットの ID と SEQ の組を比較し，すでに受信済みのパケットか否かを調べる．すでに受信済みのパケットであれば，パケットを破棄して処理を終了する．キャッシュ上に存在しない ID と SEQ の組であれば，初めて受信したパケットであると判断し，手順 3 を行うと同時にキャッシュの更新も行う．手順 4 では，中継の最終段階としてパケットの送信を行う．

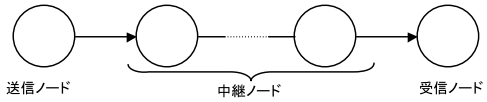

図 11 評価用トポロジ
Fig. 11 Topology for evaluation.

4. 性能評価

提案手法に基づき，実機を用いてプロトタイプシステムを作成し，その性能評価を行った．実装対象となる計算機の性能は，3 章で述べたとおりである．通信速度は，IEEE802.11b のブロードキャスト通信であるため 1Mbps となる．また，従来手法であるアプリケーションによる実装手法とカーネルの IP 層による実装手法を実装し，それらを比較対象とした．以下，本論文では，アプリケーションによる実装手法とカーネルの IP 層による実装手法をそれぞれ従来手法 1，従来手法 2 と記す．

性能評価は，**図 11** に示す構成のネットワークを構築し，中継ノード数を必要に応じて変更しながら行った．測定値に関しては，正確な時刻同期が困難であることと電波伝搬の影響を考慮し，1,000 回通信を往復して行い，送信ノードにおけるターンアラウンドタイムを 2 で割った値の平均値を採用した．

4.1 中継遅延とその詳細

中継遅延は，中継ノードが存在しない 1 ホップ通信と 1 つの中継ノードが存在する 2 ホップ通信の通信時間の差で求めることができる．各手法における通信時間と中継遅延は，**図 12** に示すとおりであった．ただし，実験環境をすべて同一条件にすることが困難であったため，平均値を用いた場合，MAC アクセス制御によって発生する遅延に関して，実験ごとに相違が見られた．したがって，公平性のため，実験結果として最も MAC アクセス制御による遅延時間が小さい場合の結果である最速値を採用した．図 12 から分かるように，中継処理を行う階層が低いほど中継遅延が小さいという結果が得られた．

図 11 のホップ数を 1〜4 ホップまで増加させた場合の通信時間を計測した結果は，**図 13** のとおりとなっ

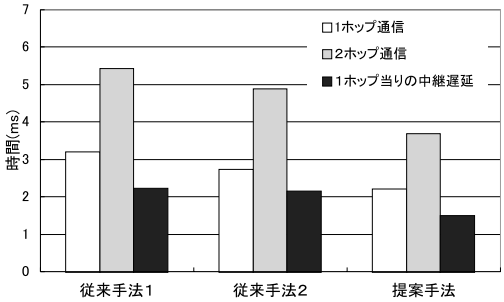

図 12 各手法の通信時間と中継遅延
Fig. 12 Communication time and relay latency of each methods.

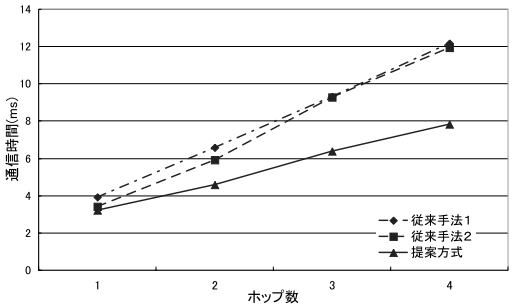

図 13 ホップ数と通信時間の関係
Fig. 13 Relationship between number of hops and communication time.

表 1 ソフトウェア処理による中継遅延
Table 1 Relay latency by software processing.

	従来手法 1	従来手法 2	提案手法
中継遅延	0.97 ms	0.95 ms	0.35 ms

た．トポロジが直線であるため，ホップ数と通信時間は，ほぼ線形に増加している．したがって，提案手法を用いた場合の通信時間は，中継遅延を削減しているため，ホップ数が増加するほどその効果が大きくなることが分かる．

中継ノードにおける中継遅延のうち，ソフトウェア処理における遅延時間は，**表 1** のとおりであった．この測定では，デバイスドライバの受信割込みハンドラの入口と送信命令書き込み直前の 2 カ所において CPU のクロック数を取得し，その差分を用いた．測定の結果，2 つの従来手法の差は，0.02 ms と非常に小さかった．一方，提案手法は，従来手法に比較して約 0.6 ms 小さい遅延時間で処理できており，70%を超える改善率を示している．

従来手法 1 と 2 との差は，図 4 と図 5 に示したとおり，UDP 層の処理およびアプリケーションとカーネル間でのコンテキストスイッチの有無である．従来手法 2 と提案手法との差は，プロトコルスタック処理の有無と，遅延処理の有無である．**図 14** は，表 1 に示した処理遅延の構成を示したものである．2 つの従来手法間では，IP 層と UDP 層以上の部分以外は，共通であるため，同じ値となっている．従来手法 2 では，従来手法 1 に比べ UDP 層とアプリケーション処理がなくなっているが，中継処理が IP 層に移されているため，IP 層での処理時間が増加している．提案手法は，遅延処理および上位プロトコルによる処理をすべて省いているため，デバイスドライバによる遅延時間のみとなっている．ただし，中継処理をデバイスドライバ内で行うため，その分だけ従来手法と比べてデバイスドライバでの遅延時間が大きくなっている．従来手法 2 の IP 層における遅延時間の増加分より，提案手法の増加分の方が大きくなっている．これは，提案手法がアプリケーションと直接パケットをやりとりするためのキューを持っており，そのキュー操作によるものと考えられる．

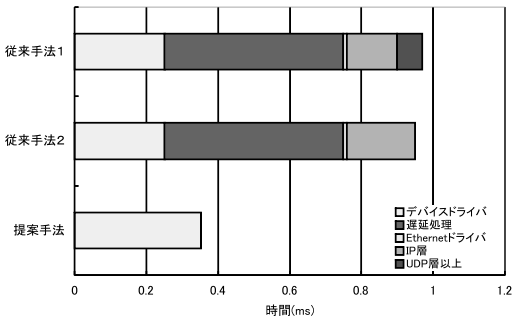

図 14 中継遅延の内訳
Fig. 14 Detail of delay of relay.

表 2 送信時と受信時の処理時間
Table 2 Processing time of transmission and reception.

	従来手法 1	従来手法 2	提案手法
送信時	0.23 ms	0.23 ms	0.20 ms
受信時	1.82 ms	1.81 ms	1.40 ms
合計	2.05 ms	2.04 ms	1.60 ms

4.2 送受信の高速化の効果

提案手法では，通信プロトロコルとして UDP/IP を用いずに通信することで，中継時だけでなく送受信の高速化を図っている．送信時と受信時のソフトウェアにおける処理時間は，**表 2** に示すとおりであった．

測定の結果，送信時において約 0.03 ms，受信時において約 0.4 ms 高速化され，全体として約 0.45 ms の遅延が削減されたことが分かる．受信時における効果が送信時のそれよりも大きい理由は，受信時において行われる遅延処理を削減したことによる．

4.3 考　　察

図 12 と図 13 から，提案手法により，フラッディングの中継遅延を約 30% 削減することが可能であり，その効果はホップ数が増えるほど高くなることが分かる．提案手法が対象とするパケット受信から中継処理を経てパケットを送信するまでのソフトウェアによる処理遅延は，表 1 に示したとおり，約 70% の高速化が達成できた．ここで，図 12 で示した提案手法と従来手法の中継遅延の差は，表 1 の中継遅延の差である．これらの結果から，フラッディングの高速化において，提案手法が有効であることが実証できた．

提案手法により削減された遅延は，図 14 のとおりであった．この結果から，従来手法における遅延の 50%超は遅延処理によるものであることが分かる．遅延処理は，様々な OS に採用されている一般的な手法である．したがって，提案手法は，プロトタイプシステムで用いた Linux 以外でも有効であると考えられる．

提案手法では，3 GHz の CPU を搭載した計算機を用いている．プロトコルスタックの処理遅延等は，計算機性能によるところが大きい．そこで，クロック数が 700 MHz の CPU を搭載し，OS に Fedora3 Linux（kernel2.6.9）を用いた計算機において，遅延処理による遅延がどの程度変化するかを計測した[11]．その結果，パケット受信時における遅延処理による遅延時間は，高速な CPU を用いているプロトタイプシステムとほぼ同じ約 0.49 ms であった．このことから，提案手法は，遅延処理による遅延に関しては，計算機性能によらず一定の効果が期待できると考えられる．遅延処理による遅延は，中継処理において大きな割合を占める．これらから，提案手法は，計算機性能によらずフラッディングの高速中継の実現に有効であるといえる．

5. 関 連 研 究

従来のアドホックネットワークにおけるフラッディングに関する研究は，中継パケット数を減少させつつ高いパケット到達率を実現するアルゴリズムに関するものが行われてきた．高速パケット中継を目的とした研究では，標準化が進められている IEEE802.11s とハードウェアによる中継技術に関して行われている．以下，本章では，フラッディングアルゴリズム，IEEE802.11s とハードウェアによるパケット中継技術について述べる．

5.1 フラッディングアルゴリズム

フラッディングでは，ブロードキャストアドレスに対してパケットを送信する．802.11MAC プロトコルでは，ブロードキャスト時に RTS/CTS を用いない．したがって，フラッディング時には，衝突の発生頻度が上昇する可能性が高い．また，フラッディングでは，ノード密度が高いと中継ノード数が増えるため，輻輳状態を引き起こしやすいという問題がある．これらの問題を解決するため，フラッディングアルゴリズムに関する研究が行われてきた．これらの研究では，フラッディング時に中継を行うノード数を減らし，ネットワーク上に送信されるパケット数を減らしつつ，高いパケット到達率を実現することを目的としている．

フラッディングにおいて最も基本となるアルゴリズムは，ピュアフラッディングである．ピュアフラッディングでは，フラッディングパケットを受け取ったすべてのノードが中継を行う．中継ノードは，1 回中継したパケットをキャッシュしておくことで，冗長な中継を避けている．その他のアルゴリズムには，以下にあげる手法が提案されている．

- タイマを用いた手法[12]
- 確率を用いた手法[12]
- 位置情報を用いた手法[12]
- 近隣ノード情報を用いた手法[13]

タイマを用いた手法では，パケットを受け取るとタイマを設定する．ノードは，タイマが切れるまでの間に，中継しようとしているパケットと同一のパケットを何回受信したかを計測する．この計測結果を基に，ノードは，中継を行うか否かを判断する．確率を用いた手法では，ノードにあらかじめ一定の中継確率を与えておく．ノードは，この確率に基づいて中継を行うか否かを判断する．位置情報を用いた手法では，自ノードと直前の中継ノードとの間の距離が，ある範囲内にあるときに限り中継を行う．または，中継を行うことでより広い範囲にパケットを行きわたらせることができると判断したときに中継を行う．ただし，この手法では，電波強度に基づく距離推定機能や GPS 等の測位デバイスを使用することを前提としている．近隣ノード情報を用いた手法では，2 ホップ圏内程度の近隣ノード情報に基づいて，次に中継すべきノードを送信ノードまたは中継ノードが指定する．ただし，近隣ノード情報は，フラッディングを行う前に獲得されていなければならない．したがって，各ノードは，Hello パケット等を用いて定期的にノード情報を更新する必要がある．

提案手法は，これらのフラッディングに関する研究と異なり，フラッディングの中継時における中継処理をその構成手法により高速化することを目的としている．したがって，これらのフラッディングアルゴリズ

ムと提案手法は，同時に使用することが可能であり，それにより高速かつ効率的なフラッディングの実現が可能になると考えられる．ただし，提案手法が想定している移動性のある環境やアルゴリズムの複雑さを考慮すると，これらのアルゴリズムを改良する必要がある．

5.2 IEEE802.11s

IEEE802.11s は，メッシュネットワークに関する規格であり，MAC 層にパケット中継機能を実現することを想定している[8]．メッシュネットワークには，無線 LAN メッシュネットワーク機能のみを実装した MP（Mesh Point），無線 LAN アクセスポイントの機能を実装した MAP（Mesh Access Point），MP に外部ネットワーク接続のためのゲートウェイ機能を持たせた MPP（MP collocated with a mesh Portal），通常の無線 LAN 端末である STA（STAtion）がある．MAC 層でのパケット中継機能を持つのは，STA 以外の端末である．メッシュネットワークでは，通信を高速化するために，IP 層で IP アドレスを用いてパケットの中継先等を決めるのではなく，MAC 層で MAC アドレスを用いて中継先を決定する．フラッディングの場合も，同様に MAC 層で行われる．さらに，通信デバイスを 1 つ使う共通チャネルモードと，チャネルの有効利用やネットワークの大容量化のために，複数デバイスを利用するマルチチャネルモードがある．

MAC 層でパケット中継を行う点では，提案手法とメッシュネットワークは共通であるが，既存の実装手法では通信デバイスのデバイスドライバと IP 層の間に仮想デバイスドライバを設けており，実装位置が提案手法とは異なる．また，メッシュネットワークは，STA の移動が考慮しているものの，メッシュネットワークを構築する MP 等の移動は追加・削除程度しか想定されておらず，フラッディングアルゴリズムもピュアフラッディングに比べて移動性に弱い近隣ノード情報に基づいた手法が採用されている．したがって，提案手法が想定するノードがつねに高速移動している環境では，安定した利用が困難であると考えられる．また，メッシュネットワークが想定するマルチチャネル環境では，複数の通信デバイスが 1 つの MP に与えられるため，チャネルをまたがってパケットの中継を行う際には，ネットワーク間接続と同じ処理を行う必要があるため，完全にデバイスドライバ内で閉じた実装を行うことが困難であると考えられる．

5.3 ハードウェアによるパケット中継技術

アドホックネットワークにおける中継処理を高速化するために，中継処理を通信デバイスで処理する手法[4),5)] が提案されている．この手法は，ルータやスイッチングハブで採用されているカットスルー方式を通信デバイス内に実現する．パケットは，ソフトウェアではなくハードウェアで中継されるため，最も高速化が期待できると考えられる．しかし，ハードウェアによる中継処理を実現するためには，アドホックネットワークと有線ネットワークでは前提条件が異なるため，解決すべき問題がいくつかある．たとえば，アドホックネットワークでは，ハードウェアにより高速に中継処理ができても，チャネルアクセス権の獲得に時間がかかり，パケットの送信が即座に行えない可能性がある．したがって，従来とは異なるチャネルアクセス方式を用いて，あらかじめパケットの送信が行える状態になっている必要がある．

また，文献 4) と文献 5) では，カットスルー方式の実現に必要となる LDP（Label Distribution Protocol）[14] をソフトウェアにより実現するとしている．しかし，LDP をアドホックネットワーク上で実現する手法に関しては，言及されていない．このようなことから，通信デバイスによる通信の高速化は，有望ではあるが様々な課題をかかえているといえる．

提案手法は，既存の通信デバイスを用いている．変更を必要とする部分は，デバイスドライバの一部と，アプリケーションの送受信方法のみであり，実現が容易である．このように，提案手法は，既存システムに適用することが容易であるという利点を持つ．

6. おわりに

本論文では，アドホックネットワークにおけるフラッディングの中継遅延を削減し，通信を高速化する構成手法について述べた．提案手法は，有線ネットワークにおける中継とアドホックネットワークにおける中継の違いに着目している．有線ネットワークでは，中継の目的がネットワーク間接続であり，そのために複数の通信デバイスを必要とする．アドホックネットワークでは，中継の目的がネットワーク内接続であり，必要とする通信デバイスは 1 つである．これらの違いに基づき，提案手法では，有線ネットワークでは実現が困難であるデバイスドライバによる中継処理の実現を提案している．さらに，パケットの送受信時において，ユーザが直接デバイスドライバに要求を出せるように拡張することで，カーネルのプロトコルスタックを回避し，送受信における遅延も削減する．

本論文では，提案手法のプロトタイプを実装し，その性能評価を行うことで提案手法の効果を検証した．実験の結果，提案手法が対象とするソフトウェアによ

る中継処理において，従来手法と比較して約 70% の遅延が削減された．ハードウェアによるチャネルアクセスや電波伝搬も含めた通信時間に関しては，約 30% の削減が実現できている．提案手法により削減された中継遅延の詳細を調べた結果，遅延の要因にはプロトコルスタックによる処理遅延と遅延処理による遅延が存在することが分かった．特に，遅延処理による遅延は，従来手法の遅延時間の 50% 以上を占めており，またその遅延時間は計算機性能によらないことが分かった．したがって，提案手法は，計算機性能によらず遅延を削減することが可能である．また，提案手法は，既存の通信デバイスとシステムを変更するなく使用することができる．これらのことから，提案手法は，アドホックネットワークにおけるフラッディングの高速化に有効であると考えられる．

今後は，フラッディングを基本とした通信技術に提案手法を適用し，その効果を検討していく予定である．さらに，ユニキャスト通信等への適用を行い，提案手法が様々な用途に利用可能であることを示す予定である．

謝辞 本研究は，情報通信研究機構の研究依託「高レスポンスマルチホップ自律無線通信システムの研究開発」により実施した．

参 考 文 献

1) Lee, S.-J., Su, W., Hsu, J., Gerla, M. and Bagrodia, R.: A Performance Comparison Study of Ad Hoc Wireless Multicast ProtocolS, *Proc. 19th Annual Joint Conference of the IEEE Computer and Communications Societies* (*INFOCOM2000*), Vol.2, pp.565–574 (2000).
2) Ho, C., Obraczka, K., Tsudik, G. and Viswanath, K.: Flooding for reliable multicast in multi-hop ad hoc networks, *Proc. International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communication* (*DIALM*), pp.64–71 (1999).
3) 関 馨：海外における車車間通信の開発—その現状と現状，自動車研究，Vol.27, No.1, pp.21–26, 日本自動車研究所 (2005).
4) Acharya, A., Misra, A. and Bansal, S.: A label-switching packet forwarding architecture for multi-hop wireless LANs, Technical Report, IBM (2002).
5) Ramanathan, R.: Challenges: A Radically New Architecture for Next Generation Mobile Ad Hoc Networks, *Proc. 11th Annual International Conference on Mobile Computing and Networking, MOBICOM 2005*, ACM, pp.132–139 (2005).
6) Chakeres, I.D. and Belding-Royer, E.M.: AODV Routing Protocol Implementation Design, *Proc. 24th International Conference on Distributed Computing Systems Workshop*, pp.698–703 (2004).
7) netfilter/IPtables org: netfilter/iptables project. http://www.netfilter.org/
8) 阪田史郎，青木秀憲，間瀬憲一：アドホックネットワークと無線 LAN メッシュネットワーク，電子情報通信学会論文誌 B，Vol.J89-B, No.6, pp.811–823 (2006).
9) Draves, R., Padhye, J. and Zill, B.: Routing in Multi-Radio, Multi-Hop Wireless Mesh Networks, *Proc. 10th annual ACM/IEEE International Conference on Mobile Computing and Networking*, pp.114–128 (2004).
10) Lilieblad, F., Lundberg, D., Nielsen, J., Nordstrm, E. and Tschudin, C. F.: Mad-hoc AODV Implementation and Documentation. http://mad-hoc.flyinglinux.net
11) 瀧本栄二，滝沢泰久，門脇直人，小花貞夫，大久保英嗣：アドホックネットワークにおける中継遅延の削減方式，*FIT2006*, p.L-075 (2006).
12) Ni, S., Tseng, Y., Chen, Y. and Sheu, J.: The Broadcast Storm Problem in a Mobile Ad Hoc Network, *Proc. International Conference on Mobile Computing and Networking, MOBICOM 1999*, pp.151–162 (1999).
13) Qayyum, A., Viennot, L. and Laouiti, A.: Multipoint Relaying: An Efficient Technique for flooding in Mobile Wireless Networks, Technical Report Research Report RR-3898, INRIA (2000).
14) IETF: RFC3036 LDP. http://www.ietf.org/rfc/rfc3036.txt

(平成 19 年 5 月 18 日受付)
(平成 19 年 11 月 6 日採録)

瀧本 栄二（正会員）

1999 年立命館大学理工学部情報学科卒業．2001 年同大学大学院理工学研究科博士課程前期課程修了．2004 年同大学院理工学研究科博士課程後期課程単位取得退学．同年 ATR 適応コミュニケーション研究所入社．無線アドホックネットワークおよび車車間通信に関する研究に従事．

滝沢　泰久（正会員）

1983 年京都工芸繊維大学工芸学部機械工学科卒業．同年日本ユニシス（株）入社．1990 年住友金属工業（株）入社．1998 年 ATR 環境適応研究所出向．2002 年（株）国際電気通信基礎技術研究所適応コミュニケーション研究所主任研究員．現在，無線ネットワークにおける自己組織化等の研究に従事．工学博士．電子情報通信学会，IEEE，IEEE-CS 各会員．

鈴木龍太郎

1977 年電気通信大学電気通信学部物性工学科卒業．1979 年同大学大学院修士課程修了．同年郵政省電波研究所（現 NICT）に入所し衛星通信の研究に従事．1992 年文部省放送教育開発センター助教授，1996 年 CRL 通信システム部非常時通信研究室長．1997 年 TAO 川崎次世代 LEO リサーチセンターサブリーダー．2002 年 CRL 無線通信部門先進衛星技術実証グループリーダー．2004 年，高速衛星ネットワークグループリーダー．2007 年より株式会社 ATR 適応コミュニケーション研究所スマートネットワーク研究室長として無線ネットワークの研究開発に従事．IEICE，IEEE，AIAA 各会員．IAF Satellite Communication and Navigation Committee 委員．博士（工学）

小花　貞夫（フェロー）

1976 年慶應義塾大学工学部電気工学科卒業．1978 年同大学大学院修士課程修了．同年国際電信電話（株）（現 KDDI（株））入社．パケット交換方式，ネットワークアーキテクチャ，OSI プロトコル実装，データベース，ビデオテックス，分散処理，ネットワーク管理，ITS の研究・開発に従事．2004 年（株）国際電気通信基礎技術研究所適応コミュニケーション研究所所長，アドホックネットワーク，ITS，センサネットワークの研究開発に従事．工学博士．2001 年文部科学大臣賞（研究功績者），本会フェロー，電子情報通信学会会員．