NAIST-IS-MT9751112

修士論文

# 広域無線通信メディア利用時のゲートウェイ協調による TCP の性能改善

山本 一隆

1999年2月12日

奈良先端科学技術大学院大学
情報科学研究科 情報システム学専攻

本論文は奈良先端科学技術大学院大学情報科学研究科に
修士 (工学) 授与の要件として提出した修士論文である。

山本 一隆

指導教官： 湊 小太郎 教授
横矢 直和 教授
砂原 秀樹 助教授

# 広域無線通信メディア利用時のゲートウェイ協調による TCP の性能改善*

山本 一隆

## 内容梗概

携帯電話や PHS などの広域無線通信メディアを利用してインターネットに接続を行う場合、無線状況による RTT の揺らぎ、データリンク層とトランスポート層でのデータの信頼性保証の重なり等の影響により、データの転送効率が著しく低下するという問題がある。

本稿では、ゲートウェイ協調により、広域無線通信メディア利用時の TCP の性能を向上させることを提案する。協調ゲートウェイは TCP コネクション上で、データの信頼性を保証するリンク層と保証しないリンク層の間に介在し、受信ウインドウを考慮した ACK の生成を行うことによって、送信側のデータ転送をコントロールする。これにより無線状況悪化時のデータの転送効率の低下を防ぐことが可能となる。実験の結果、無線状況悪化時のデータ転送速度の改善が確認された。

キーワード

TCP, 広域無線通信メディア, ゲートウェイ, 代理 ACK, 受信ウインドウサイズ

*奈良先端科学技術大学院大学 情報科学研究科 情報システム学専攻 修士論文, NAIST-IS-MT9751112, 1999年2月12日.

# Improving TCP Performance over Cellular Telephone Network by Cooperative Gateway*

Kazutaka Yamamoto

**Abstract**

TCP cannot work effectively over cellular telphone network because of fluctuation of RTT. Error control schemes of cellular telphone network also cause bad effects on TCP.

In this paper, we propose a technique, which we call cooperative gateway, to improve TCP performance in such environment. It is located between the two kinds of links, the one has error control schemes and the other doesn't. The cooperative gateway generates pseudo acknowledgments and manages sender's data transmission by controlling window size in TCP header. Our experiments show that cooperative gateway improves TCP performance.

**Keywords:**

TCP, Cellular Telephone Network, Gateway, Pseudo Acknowledgement, Receive Window Size

*Master's Thesis, Department of Information Systems, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-MT9751112, February 12, 1999.

# 目 次

# 図目次

# 表目次

# Chapter 1

# 序論

## 1.1. はじめに

近年では，計算機を取り巻く環境としてインターネットが普及し、その利用者も急激に増加してきた。従来のインターネットへの主な接続形態としては、構内の固定ホストで構成されたLANを介しての接続、家庭内の固定ホストからの公衆電話網を介したダイアルアップでの接続が主であった。こうした接続は有線ネットワークを用いた接続である。しかし、ここ数年で爆発的に普及した携帯電話、PHSなどの広域無線通信メディアや、無線LAN、人工衛星を利用した衛星回線などの無線ネットワークによる接続という選択肢が出てきた。このような無線ネットワークを利用することによって、有線で継った環境と異なり、移動しながら、あるいは移動先からネットワークに接続し計算機で情報処理を行うというモバイルコンピューティングが可能になった。

しかし、現在のインターネットで広く利用されているTCP/IPでは様々な理由から無線環境で十分に適応出来ないという問題が過去の研究により指摘されている[1]。これらのプロトコルは本来イーサネット、FDDI等の伝送速度で、誤り率が小さい通信メディアでの使用を前提とし，通信するホストの移動は前提とはしていなかった。つまり、無線ネットワークのように低帯域、高い誤り率、データリンク層で独自の誤り訂正、通信が断続的になるといった有線環境とは異なるメディアでの利用を考慮して設計されたものではなかった。そのために、インター

ネットにおいて無線ネットワークを介しての通信では、有線環境とは異なるメディアの特徴がTCPに悪影響を及ぼして、スループットの低下や不必要な再送などの問題を起こしてしまう。

今後のモバイルコンピューティングの発展には、無線ネットワークによる問題点の解決が必須であり、無線ネットワークを考慮したプロトコルの設計が大きな課題だと言える。またその際に既存技術を可能な限り有効に活用するということが重要になってくる。システムの根本的な改良を行うとすると莫大なコストを要することとなるので、できるだけ既存のシステムの変更は少なくしなければならない。

本研究では無線ネットワークの中でも携帯電話、PHSなどの広域無線通信メディアを利用したTCPに注目する。広域無線通信メディアではデータリンク層独自の誤り訂正をもつ。このデータリンク層での誤り訂正を有効に利用するため、有線区間、無線区間を分けるゲートウェイに注目し、そのゲートウェイのみに改良を加え性能改善を図る手法について提案する。

## 1.2. 本論文の構成

2章ではインターネットでの信頼性のあるデータ配送機能を提供するプロトコルであるTCPの機能について述べる。3章では無線通信メディアの特徴を述べた後、無線通信メディアを利用したTCPの問題点についての検証を行う。4章では本研究の目的である広域無線通信メディア利用時のゲートウェイ協調によるTCPの性能改善についての方針につて述べ、5章でその具体的な提案、設計を行う。6章では設計に基づいた協調ゲートウェイの実装、評価について述べる。7章、8章で今後の課題、結論を述べる。

# Chapter 2

# TCP の機能

急速に発展しているインターネットを動かしている最も重要な技術は TCP/IP と呼ばれるプロトコルである。TCP/IP とは TCP(Transmission ControlProtocol)[2] と IP(Internet Protocol)[3] という二つの異なった役割をもつプロトコルをまとめて呼ぶときに使われる。この二つのプロトコルのどちらかが欠けても現在のインターネットは成り立たない。

IP はパケットを受信ホストに届けるためにできる限りの努力をするが，届けられるかどうかは保証されない。途中でパケットが喪失したり，順序が入れ替わったり，1 つのパケットが 2 つ以上に増えることも起こりうる。

TCP は信頼性の無い IP のパケット配送機能を利用してエンドホスト間で制御を行い、信頼性のあるデータ配送機能を提供するプロトコルである。TCP はエンドホスト間でコネクションの制御、フロー制御、再送制御、輻輳回避制御を行う。

TCP を利用するアプリケーションは多い。telnet による遠隔ログインや、ftp によるファイル転送、電子メールの配送、WWW のデータの転送など、インターネットのほとんどのデータ通信には TCP が利用されている。これらの多くのアプリケーションは TCP が提供する信頼性を必要としている。

TCP はコネクション指向の信頼性のある全二重のバイトストリームを提供するトランスポートプロトコルである。コネクション指向とは、通信に先だって通信をする相手との間で仮想的な通信路を作り、その通信路を利用してデータ転送を行うことである。

信頼性の保証に関しては TCP では以下の方法で行う。

- アプリケーションデータは、TCP によって通信に最適なサイズに分割される。この分割されたデータはセグメントと呼ばれる。
- TCP はセグメントを送るときにタイマーを設定し、他方のエンドからセグメントが受信されたことを知らせる ACK(確認応答) を待つ。時間内に ACK が返送されなければ、セグメントは再転送される。2.2 節で TCP のタイムアウトと再転送の仕組みを述べる。
- TCP がコネクションの一方のエンドからデータを受信すると ACK を送り返す。
- TCP ヘッダとデータでチェックサムを利用する。これはデータが転送中に変化していないかどうかを検出するためである。セグメントを正しくないチェックサムで受信したら、TCP はそれを破棄し、ACK は返送しない (それにより送り手のタイムアウトと再転送を待つ)。
- TCP セグメントは IP データグラムとして転送される。しかし、IP データグラムは順番どおりに到着しない場合が生じる。データの受信側の TCP は、必要であればデータを順に並び替え、受信したデータを正しい順序でアプリケーションに渡す。
- IP データグラムは重複することがある。しかし、受信側の TCP は重複したデータを破棄しなければならない。

また TCP ではネットワークを有効に利用し、かつ高スループットを得るために大きくわけて次の 3 つの制御機構がある。これらは TCP の転送能力に大きく関係する。

1. **フロー制御**

   エンドホスト間で最大のスループットが得られるように制御する。

2. **再送制御**

   セグメントが喪失したかどうかをできるだけ素早く正確に予測して再送する。

3. 輻輳回避制御

   ネットワークの混み具合にあわせてセグメントの送信量を制御し、パケットの破棄を防ぐ。

この3つの制御機構について説明する。

## 2.1. フロー制御

TCP のフロー制御には受信側ホストによるものと、送信側ホストによるものとの2つある。

### 2.1.1 受信側ホストによるフロー制御

TCP はエンドホスト間のフロー制御としてスライディングウィンドウ制御を採用している。ウィンドウとは ACK が到着するまでの間に送信できるバイト数のことである。figure2.1は TCP のヘッダである。ウインドウサイズのフィールドはその時点でそのホストが受信できるバイト数である。つまり、確認応答番号フィールドで指定される番号から広告するウインドウサイズまでのデータは受信できることを意味する。受信側ホストから返送される ACK のこのフィールドを利用し、送信側ホストはデータの送信の抑制や停止、再開といったフロー制御を行う。

受信ホストが受信しているデータを十分に処理できない場合はその分小さなウィンドウサイズを広告することで、受信バッファがあふれないように送信ホストのデータ送信を抑制する。送信側ホストはウインドウサイズ0を広告された場合、データを送信することはできない。その場合、送信側ホストはタイマーで設定された時間を経過する度に、1バイトのデータを送信して、受信側ホストの受信可能ウィンドウサイズが更新されたかどうかを確認する。

### 2.1.2 送信側ホストによるフロー制御

TCP コネクションのフロー制御が受信側ホストの広告するウィンドウだけであるとしたら、送信側ホストは受信側ホストの広告するウィンドウサイズの限界

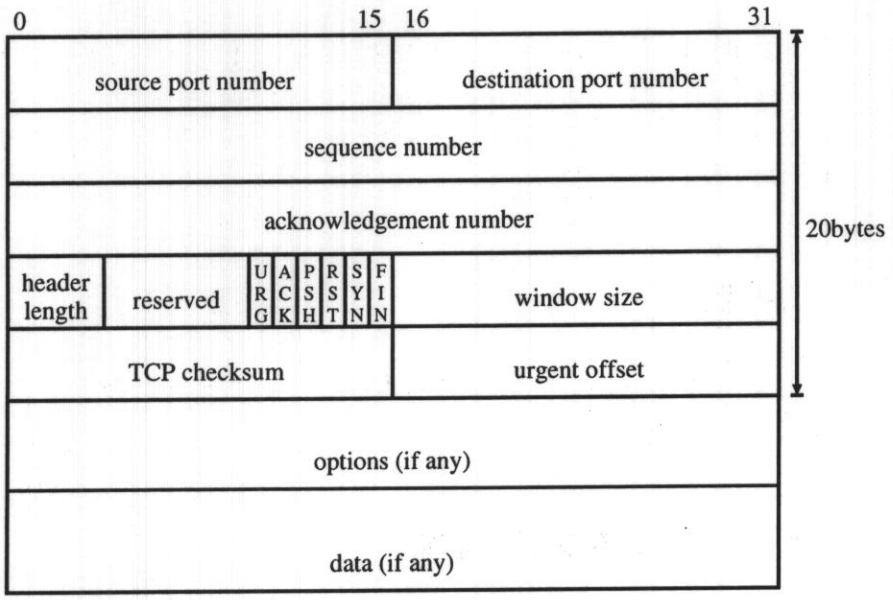


Figure 2.1 TCP のヘッダ

までデータを送信してしまう。しかし、TCP コネクションのエンドホスト間に伝送速度の小さなネットワークがあるような場合、途中のルータでデータをキューに入れて貯めておかなければならない。そのキューが溢れると輻輳が発生し、スループットを著しく低下させてしまう。このようなことを避け、ネットワークに送り出されるパケットの通信速度を、他方のエンドから返される ACK の通信速度と同期させるためにスロースタートというアルゴリズムがある [4]。これは送信側ホストが行うフロー制御である。

スロースタートは送信側の TCP にフロー制御のウィンドウとは別に輻輳ウインドウ (cwnd) と呼ばれるウィンドウを保持しておく。TCP コネクションが確立されると輻輳ウィンドウはセグメント 1 つ分に初期化される。ACK が受け取られるたびに、輻輳ウィンドウは 1 セグメントずつ増やされる。送信側ホストは、輻輳ウィンドウと広告されたウィンドウの最小値までデータを転送することが可能である。

送信側ホストは 1 つのセグメントを転送することからスタートし、その ACK を待つ。ACK を受け取ると、輻輳ウィンドウは 1 から 2 に増加され、2 つのセグ

メントが送信可能になる。それら 2 つのセグメントに対する ACK が返ってくると、輻輳ウィンドウは 4 に増加される。それは幾何級数的に増加される。

ウィンドウの大きさがある大きさまで増加した場合には、急激なデータ転送の増加によって新たな輻輳が発生しないように、スロースタートを止め、輻輳ウインドウを大きくする度合を減らす。この値はスロースタート閾値 (slow start threshold) と呼ばれる。

輻輳ウィンドウとスロースタート閾値はネットワークの輻輳具合に合わせてデータの送信量を制限し、輻輳を回避するために利用されるが、この輻輳回避の仕組みについては 2.3 節で詳しく述べる。

## 2.2. 再送制御

TCP は信頼性のあるデータストリームを提供するプロトコルである。そのため、セグメントが喪失した場合には再送を行わなければならない。セグメントの喪失は TCP コネクションの経路途中にあるルータの処理能力を越えた場合に起こることが多い。各ルータで送信できるパケット数よりも、受信するパケット数が多くなればバッファがあふれる。この現象は主にたくさんのノードがつながっているルータや、伝送速度が大きなネットワークから小さなネットワークへとパケットを中継するルータなどで発生する。しかし、通常はルータのバッファがあふれても送信ホストや受信ホストには通知されない。そのため、送信ホストや受信ホストはセグメントが本当に喪失したかどうかを知ることができない。

TCP ではセグメントが喪失したかどうかを推測することによって再送の制御を行っている。次の 2 つのいずれかの状況で、TCP ではセグメントが喪失されたと判断して再送を行う。どちらもデータを送信するホストが判断する。

1. ACK の到着が著しく遅れた場合 (タイムアウト)
2. 重複 ACK が数個 (通常 3) 以上到着した場合

この 2 つについてそれぞれ詳しく述べる。

### 2.2.1 タイムアウトによる再送

ACK の到着が遅れているかどうかを判断するために、TCP では再転送タイマーを設定するとともに、RTT(往復時間) と RTT の分散を測定している。RTT は経路の変化やネットワークのトラフィック量の変化によって、時間とともに変化する。TCP では常に RTT の変化を監視し、タイムアウトを適正な値に補正している。この設定されたタイムアウトの時間内に ACK が到着しない場合に、送信側ホストは、そのセグメントは喪失されたと判断し、再送を行う。

### 2.2.2 重複 ACK による再送

受信ホストで、次に受信すべきシーケンス番号より進んだシーケンス番号を持つセグメントが到着した場合、現在受信しているセグメントまでの ACK(重複 ACK) を送る。これはセグメントの順番が間違って送られてきたことと、期待されているシーケンス番号が何であるかを他方のホストに知らせるためである。

重複 ACK がセグメントの喪失によるものなのか、あるいは単にセグメントが入れ替わって送られてきただけなのか判断することができないため、いくつかの重複 ACK が送られてくるのを待たなければならない。セグメントが入れ替わっているいるだけなら、1 つないし 2 つの重複 ACK しか送られないはずである。そして並べ替えられたセグメントが処理されると新しい ACK が生成される。しかし、3 つ以上の重複 ACK を立て続けに受け取ったら、それはセグメントが喪失している可能性がきわめて高いことを意味している。送信側ホストでこの重複 ACK を 3 つ以上受信した場合、そのセグメントは喪失されたと判断し、再送を行う。

## 2.3. 輻輳回避制御

TCP ではセグメントの喪失が起きた場合には、常に輻輳が起きたと判断する。輻輳が起きた場合には輻輳回避のための動作をする。セグメントの喪失の判断に関しては 2.2 節でみたが、それぞれに対しての輻輳回避処理が異なる。

この節では、2.1 節で触れた輻輳回避のアルゴリズムの詳細について述べ、次

に輻輳が実際に起きた後の処理についてセグメントの喪失の判明別に述べる。

### 2.3.1 輻輳回避アルゴリズム

スロースタートではACKが返ってくるたびに指数関数的に輻輳ウィンドウの大きさが増加していった。しかし、輻輳ウィンドウがスロースタート閾値を越えた場合には、輻輳が発生するのを防ぐために輻輳回避アルゴリズムを適用する。具体的には1つのACKごとに(1/輻輳ウィンドウの大きさ)セグメント分ずつ輻輳ウィンドウを拡大していく。スロースタートに比べウィンドウの拡大率を小さくすることで、ネットワークに流し出すセグメントの数を抑制する。

なおコネクション確立直後は輻輳ウィンドウは1セグメント分に設定され、スロースタートが行われる。また、コネクション確立直後はスロースタート閾値は最大ウィンドウよりも大きく設定され、途中輻輳が起きない限りは、最大ウィンドウになるまでスロースタートが続けられる。

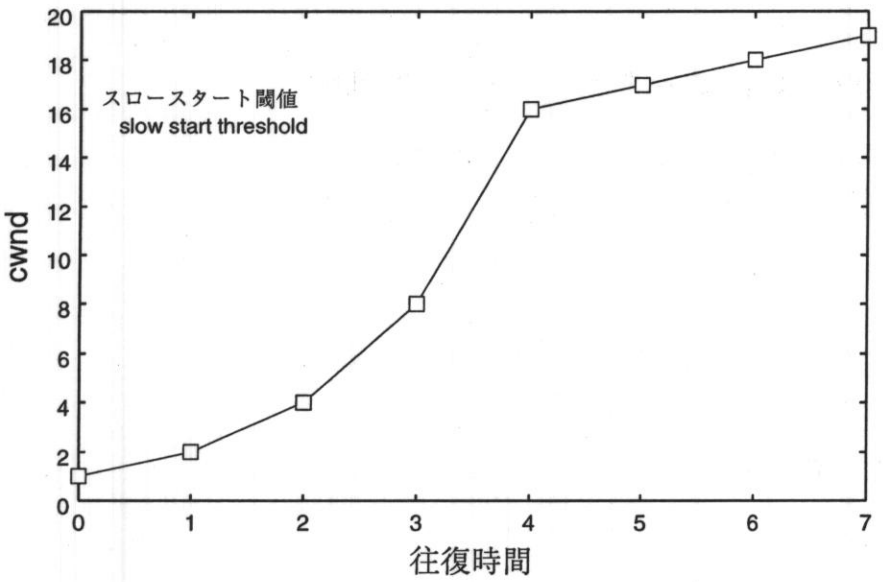


Figure 2.2 スロースタートと輻輳回避のグラフ化

figure2.2はスロースタートと輻輳回避をグラフ化したものである。cwndとssthresh-

old の単位はセグメントで示しているが、実際はバイトで扱われる。このグラフでは ssthreshold は 16 セグメントに設定されていて、時間 4 以降は輻輳回避が起動されていることを示している。

### 2.3.2 タイムアウトによる輻輳の判明

タイムアウトによりセグメントの喪失が判明した場合、輻輳が起きている可能性が高いと考え、ネットワークに流すパケット数を急激に小さくする。それは以下の手順で行われる。

1. スロースタート閾値を現行ウィンドウサイズの 2 分の 1(受信側ホストの広告したウィンドウサイズと輻輳ウィンドウの最小値、ただし少なくとも 2 セグメント) に設定する。
2. 輻輳ウィンドウの大きさを 1 セグメントに設定する。
3. 以後、スロースタート、輻輳回避の適用

### 2.3.3 重複 ACK の受信による輻輳の判明

重複 ACK の受信によりセグメントの喪失が判明した場合、セグメント喪失以外にも順番誤りのセグメントが到達しているという意味も含んでいる。よってタイムアウトによるセグメントの喪失判明よりも輻輳の度合は軽度と考えることができる。ここでタイムアウトの場合のように一気に輻輳ウィンドウを 1 セグメント分まで小さくしてしまうと、データの転送効率を著しく低下させてしまう可能性がある。それを避けるために、重複 ACK の受信により輻輳が判明した場合、以下の手順を踏んで、輻輳回避を行う。

1. 3 番目の重複 ACK が受け取られると、スロースタート閾値が現行の輻輳ウインドウの 2 分の 1 に設定する。
2. 喪失されたセグメントが再転送する。
3. 輻輳ウィンドウをスロースタート閾値に 3 セグメントサイズ分を加えたものに設定する

4. 別の重複ACKが到着するたびに、輻輳ウィンドウをセグメントサイズ単位で増加させ、(新しい輻輳ウィンドウの値で制限されない限り) 新たなセグメントを転送する。

5. 新しいデータを確認応答する次のACKが到着したら、輻輳ウィンドウをスロースタート閾値 (ステップ1で設定した値) にする。

# Chapter 3

# 無線通信メディアと TCP に関する考察

近年では計算機を接続した有線ネットワークとして、インターネットが爆発的に普及している。このインターネットを支える基盤技術である TCP/IP を中心としたプロトコルは、設計されたのが約 15 年前である。設計当時はイーサネット、FDDI 等の伝送速度が早く、bit 誤り率の小さな通信メディアでの使用を前提としていた。ところが近年の携帯電話の普及などにより、それらを利用して外出先や移動中にもインターネットへの接続をするという利用形態が浸透しつつある。また、衛星回線を利用したインターネットへの接続なども可能になってきている。

これら無線通信メディアは従来のイーサネット等とは異なる特徴を持っている。帯域に関しては、有線に比較して小さな携帯電話、PHS などの広域無線通信メディア、逆に帯域の大きな衛星通信などがある。また bit 誤り率に関しては無線通信メディア全般に高いが、中にはデータリンク層で独自の誤り訂正プロトコルを持っているものもある。また通信が断続的になる、遅延が大きいなどの特徴もある。これらの有線環境とは異なる特徴のために TCP での通信では、スループットの減少や不必要な再送など、様々な問題が生じている。

本章ではまず、無線通信メディアの特徴についてまとめる。次にそれぞれの無線通信メディア利用における TCP の問題点について考察する。また、無線通信メディア利用時の TCP の性能を改善するための関連研究についても概観する。

## 3.1. 無線通信メディアの特徴

この節では携帯電話、PHSなどの広域無線通信メディア、WaveLAN、NetWaveなどの無線LAN、また衛星回線など無線通信メディア全般について、その特徴を以下の4項目についてまとめる。

### 3.1.1 ネットワークの帯域 (伝送速度)

ネットワークの帯域は各無線通信メディアで大きく異なっている。

#### 携帯電話

現在の携帯電話 (PDC:Personal Digital Cellular phone) による回線交換型のデータ通信では、9.6kbpsが一般的な伝送速度である。最近はNTT移動通信が最大28.8kbpsの伝送速度であるパケット通信サービス「DoPa」を始めた。また、CDMA方式である次世代携帯電話IMT-2000が実現されると、高速移動中で144kbps、歩行速度で最大384kbps、屋内では最大2Mbpsの伝送速度が実現される予定である。[5]

#### PHS

PHSでは、PHSを用いたデータ通信の規格であるPIAFS(PHS Internet Access Forum Standard)が利用されており、32Kbps(実データでは29.2Kbps)のデータ通信サービスが提供されている。DDIではLAP-Pと呼ばれる独自プロトコルを開発し28.8Kbpsの伝送速度が可能である。現在では、複数の回線を束ねてデータ転送を行うことによって、64Kbpsのデータ通信を実現する準備も進んでいる。

#### 無線LAN

無線LANではWaveLAN、RangeLAN、NetWave等のワイヤレスEthernetの技術が利用されており、伝送速度は約1～2Mbpsである。

#### 衛星回線

衛星を用いたインターネットサービスは実験的プロジェクト、実用サービスなどすでにいくつかのプロジェクトが開始されている。衛星回線の帯域は非常に大きく、伝送速度では数 Mbps の単位であるが、衛星を経由するため遅延時間が非常に大きいという特徴も備えている。[6]

### 3.1.2 bit 誤り率

無線通信メディアは電波を用いているが、送信機から受信機までに伝搬損失が生じるため、有線環境に比べ bit 誤り率は高い。bit 誤りのパターンとしては、熱雑音等により時間的に不規則に発生するランダム誤りとフェージング等により集中的に発生するバースト誤りの 2 種類に大別される。

信号を伝送する途中でほとんどの場合不要な信号が混入する。このためどのような信号の処理方法をとっても望ましくない妨害を受ける。これを雑音と呼んでいる。雑音信号源はいろいろある。これらの雑音は

1. 人工雑音
2. 不規則に生じる自然界の変動性雑音
3. 物理系の中で生じるゆらぎ雑音

などに分類できる。1 は接触不良の接点、各種電気製品、点火装置、蛍光灯などの雑音源から不要な信号を拾うことにより発生する。2 は稲妻、磁気あらし、銀河系からの雑音、一般的な大気の変動などにより発生する。3 も人工的でない雑音で、抵抗内の自由電子の熱運動、真空管の電子放出、半導体のキャリア (ホールや電子) の不規則な発生、再結合、拡散などの自然なゆらぎによって物理系の内部で発生する。

figure3.1は移動通信の電波伝搬経路を模式的に書いたものであり、送信点から妨害なしに到達する直接波、建物の壁面などによる反射を経て到達する反射波、建物の屋上等から回折して到達する回折波の 3 種類の電波が受信点で受信される。これらの電波は異なる距離の伝搬路を通って受信機に到達するため、到達する信号の位相が異なってくる。受信点で受けとる信号はいろいろな経路を経た全

ての到達信号の和になる。受信端末の移動や障害物のの移動などで時間と共に、受信機に到達する信号の位相や合成信号の強さはランダムに変化する。この現象をフェージングと呼ぶ。フェージングによる急激な受信レベルの変動は通信の品質を低下させ、bit 誤り率を大きくしてしまう。

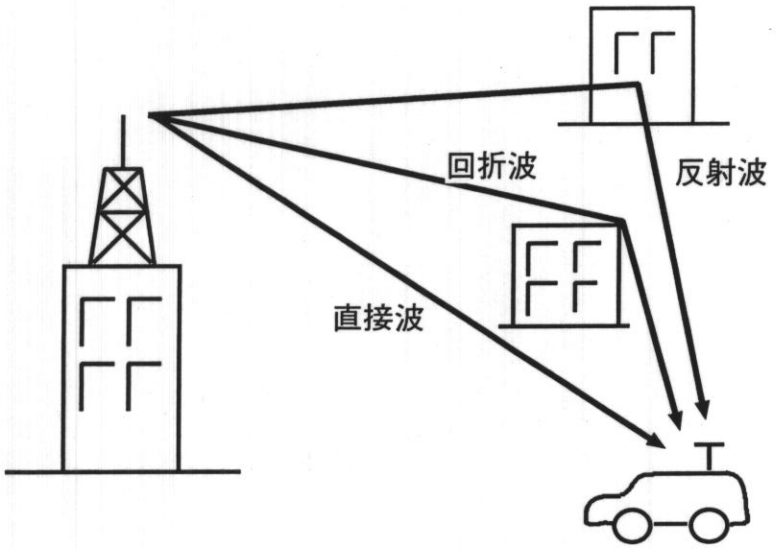


Figure 3.1 移動通信の電波伝搬経路

### 3.1.3 ハンドオーバ

無線 LAN、携帯電話や PHS などの広域無線通信メディアでは、figure 3.2のようにセルラー方式と呼ばれる通信方式をとる。この方式では基地局 (BS:Base Station) と呼ばれる無線通信のための中継地点になる機器がセルと呼ばれる通信可能範囲を持ち、セル内での移動及び通信を管理している。1つの基地局が収容できる通信チャネル数、カバーできるエリアは限界があり、この基地局を複数地点に配置することで大きなエリアをカバーしている。

なお、1つのセル半径であるが携帯電話で 1 ～ 2km、PHS で 200 ～ 500m となっている。このセルを移り変わることにより通信の継続を行うのだが、このときに以前にいたセルの基地局か次の移動先のセルに引き渡す処理をハンドオーバまた

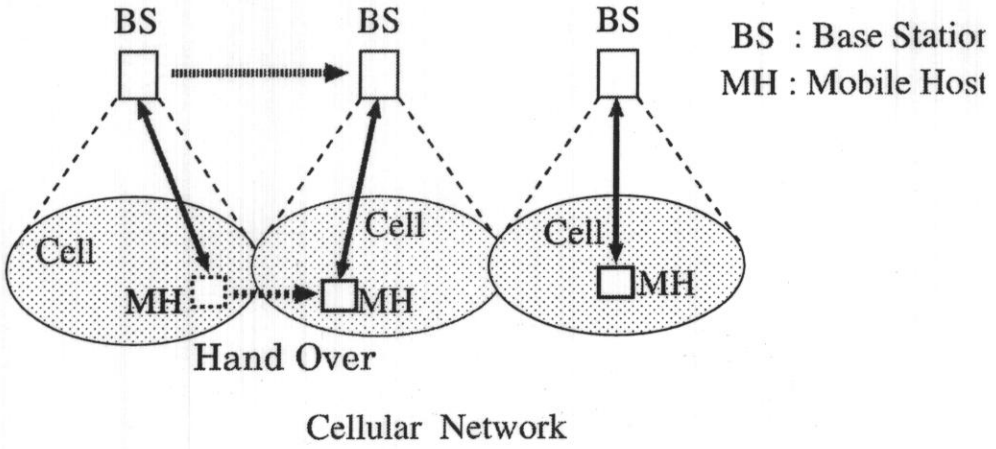


Figure 3.2 セルラー方式のネットワーク

はハンドオフといい、この間通信は一時的に中断される。次節で TCP/IP を用いた通信でのハンドオーバによるスループット低下の問題について詳しく述べる。

### 3.1.4 誤り制御

3.1.2 でみたように無線通信メディアの場合、有線通信メディアに比較して伝送品質が落ちるため、誤り制御技術による伝送品質の向上が非常に重要となる。代表的な誤り制御技術としては、順方向誤り訂正方式 (FEC:Forward Error Correction) と、自動再送要求方式 (ARQ:Automatic Repeat reQuest) に大別される。

**FEC**

FEC は、誤りが発生することを前提として、伝送すべき情報にあらかじめ冗長な情報を送信側で付加して送信することにより、無線区間で誤りが発生しても受信側のみで一方向に正しい情報を復元することができる方式である。FEC は、音声通信のようにリアルタイム性が重視される通信システムにおいて適用されるが、誤りが誤り訂正符号の能力以上に発生した場合には、エラーフリー伝送にはならない。

### ARQ

ARQ 方式は、誤り検出符号により誤りを検出したときに再度同一の情報を送出し直すことにより誤りの回復を図るものである。ARQ では、受信側から送信側へなんらかの再送要求を返すために、帰還通信路を必要とする。従って、データ伝送や画像処理等のように遅延時間のばらつきは許容されるがエラーフリーが要求されるような非電話系システムに適用される。誤りが大きい場合にもエラーフリーは実現されるが、再送回数の増加に伴いスループットは低下する。

ARQ には基本方式として次の 3 方式がある。

1. Stop and Wait ARQ(SW 方式)

   SW 方式では、フレームごとに受信局からの送達確認が返ってくるまで、新規フレームを送信しない。制御は最も簡易であるが効率が悪い。

2. Go Back N ARQ(GBN 方式)

   GBN 方式では、受信局からの送達確認を待たずに新規フレームを連続的に送信することができる。誤りフレームの再送要求が届いた場合、当該フレーム以降に送信しているフレームを再度送信するため、制御が簡便である半面、伝送効率が落ちる。

3. Selective Repeat ARQ(SR 方式)

   SR 方式も、受信局からの送達確認を待たずに新規フレームを連続的に送信することができる。SR 方式は、誤りフレームのみを再送する点で GBN 方式と異なる。最も効率がよい方式である半面、制御が若干複雑になる。

携帯電話 (PDC) におけるデータ転送では、WORM-ARQ(ARQ with Window control Operation based on Reception Memory) 方式を採用している。この方式は、高効率な SR-ARQ とバッファメモリ管理が容易な GBN-ARQ を誤り状況に応じて適応的に切替えて利用する方式である。

PHS においても ARQ は非常に効果的であり、PDC で用いられている WORM-ARQ よりも高いスループットが得られる MODS-ARQ(Modulo Operation using Data field Selective repeat ARQ) 方式を開発している。[7] [8]

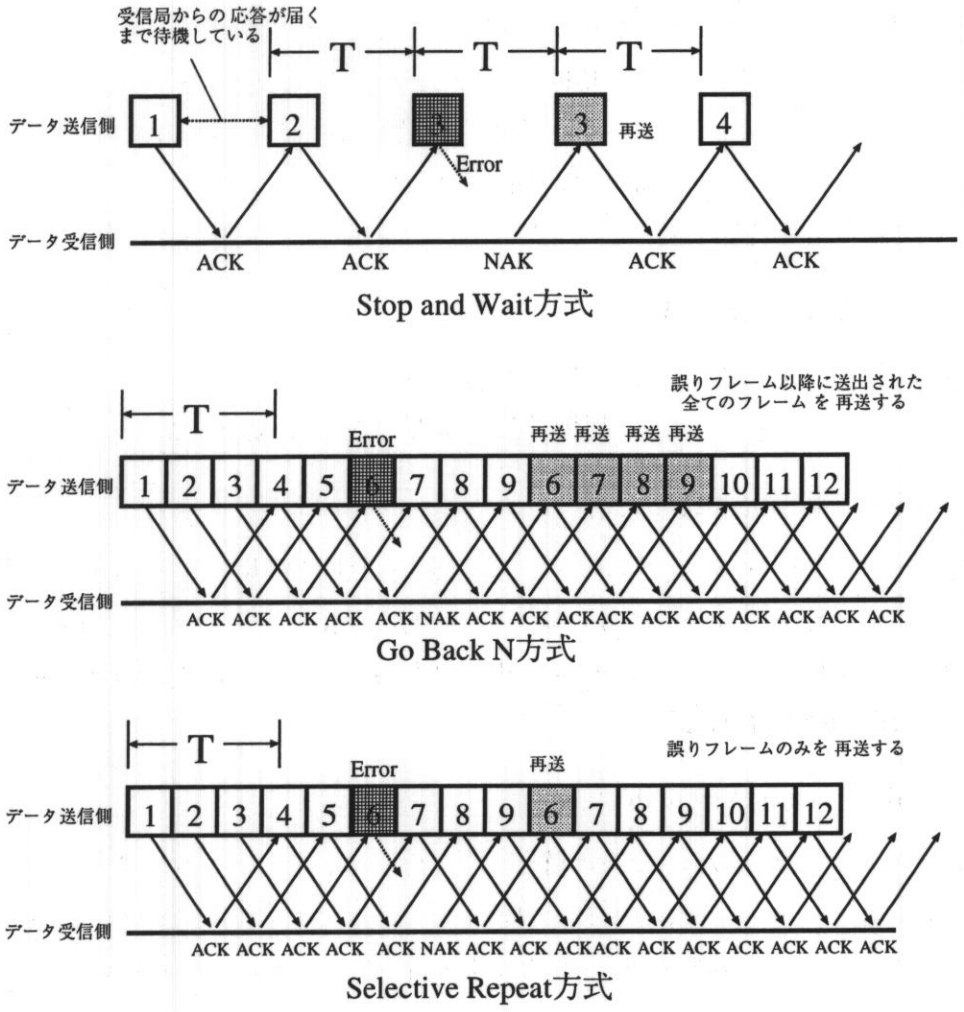


Figure 3.3 ARQ の基本的な方式

## 3.2. 無線通信メディア利用時の TCP の問題点

2 章でみたように、TCP は有線環境、固定ホストからなるネットワーク向けに設計された。このような環境のネットワークではパケットロスや異常な遅延は輻輳によって引き起こされている可能性が高い。そのため TCP は輻輳によって起きるパケットロスに対しては効率良く機能する。しかし、前節で見たように有線環境とは異なる特徴をもつ無線環境においては、輻輳以外の原因でパケットロスが起きる。このような場合、TCP ではスループットの減少や不必要な再送などの問題が生じる。

この節ではその問題点について詳しく検証する。

### 3.2.1 無線 LAN 利用時の TCP の問題点

TCP との不都合を生じる無線 LAN の特徴として、

1. 有線環境に比べ、ビット誤り率が高い。

2. セル間の移動によるハンドオーバが起きる。

が挙げられる。

ビット誤りの起きたパケットが受信側ホストに到達すると、そのパケットはチェックサムエラーにより破棄される。そのようなパケットロスがタイムアウトや重複 ACK の受信により判明すると、送信側ホストではネットワークで輻輳が起きていると判断し、輻輳回避の処理として転送速度を減少させる。しかし、実際は輻輳が起きているわけではないので、送信側での輻輳回避の処理は転送効率の低下をもたらす。

また、セルをまたぐ移動に伴いハンドオーバが起こると、基地局間の引き継ぎの処理により瞬断が起きる。この間に送出されたパケットは連続的に喪失される。TCP ではパケットロスの判明が起きる度にネットワークに送出するデータ量を減少させるので、連続的なパケットロスが起きると、転送効率が急激に悪化する。[10]

### 3.2.2 衛星回線利用時の TCP の問題点

TCP との不都合を生じる衛星回線の特徴として、

1. 有線環境に比べ、ビット誤り率が高い。

2. 衛星を経由するため遅延時間が非常に大きい。

が挙げられる。

3.1.1 で述べたように衛星回線の帯域は非常に大きく、伝送速度では数 Mbps の単位であるが、衛星を経由するため遅延時間が非常に大きいという特徴がある。衛星回線を利用した TCP では伝送遅延の影響により、データ転送速度が著しく低下してしまう。これは、TCP のフロー制御で受信バッファのオーバーフローを抑制しているため、受信側の広告したウィンドウサイズ以上のデータを送信できないからである。ウィンドウサイズ分のデータを送信するとその ACK が返送されるまで次のデータが送信できないため、伝送遅延の増加により単位時間に送信可能なデータ量が減少することになる。

また TCP のスロースタート、輻輳回避は相手の受信可能帯域が小さいことを想定し、かつネットワークの輻輳を避けるために行われる手順である。このスロースタートのために、TCP では通信開始直後は回線の容量を最大限に使用することはしない。ここでも衛星回線を利用した場合には、伝送遅延が大きいために伝送効率が低い状態から抜け出すまでに余計に時間を必要とし、効率が悪くなる。[6] [11]

無線 LAN 同様にビット誤り率の高さも問題である。移動体衛星通信システムの場合、ビット誤り率は $1 \times 10^{-4}$ 程度で、固定通信の場合にも $1 \times 10^{-6} \sim 1 \times 10^{-10}$ 程度である。これらのビット誤りによりパケットが破棄された場合にも、TCP では輻輳のために喪失されたと判断し輻輳制御を起動させる。これもデータの転送効率を低下させる原因となる。

### 3.2.3 広域無線通信メディア利用時の TCP の問題点

TCP との不都合を生じる広域無線通信メディアの特徴として、

1. 有線環境に比べ、ビット誤り率が高い。

2. セル間の移動によるハンドオーバが起きる。

3. データリンク層での誤り訂正による遅延の揺らぎが大きい。

が揚げられる。

有線環境に比べビット誤り率が高いのは無線LAN、衛星回線と同じである。しかし、3.1.4で述べたように、広域無線メディアを利用した場合には、データリンク層での誤り訂正プロトコルでエラーフリーが実現されている。そのため無線状況が悪い場合には伝送遅延の揺らぎが大きい。この伝送遅延の揺らぎの大きさがTCPの挙動に大きな影響を与える。

figure3.4はそのTCPの挙動を示したものである。なおここでは、信頼性を保証しないデータリンク層のメディアでつながっているホスト(以下、有線側ホスト)から、信頼性を保証するデータリンク層のメディアでつながっているホスト(以下、無線側ホスト)へのデータ転送を考慮している。

ゲートウェイを通過したパケットは、無線状況により無線側ホストへの到達が遅れる。figure3.4では2番のパケットの到達が大きく遅れているのがわかる。この時、無線側ホストからのACKは返ってくるのが遅れる。TCPではタイムアウトによりパケットロスの判断をしている。このタイムアウトの時間はパケットを送信して、その確認応答が返ってくるまでの時間であるRTTを基に、計算されている。順調に行われていた通信が無線状況の悪化により伝送遅延が大きくなり、受信側からのACKの返送が遅れると、TCPでタイムアウトを起こしパケットの再送、輻輳ウィンドウの減少が起きる(figure3.4のtimeout)。しかし、無線リンク層においてエラーフリーは保証がされているので、以前送信したパケットは無線側ホストに確実に届き、確認応答が行われる。従って、再送されたパケットは無線側ホストにとっては以前に受信したデータであり、期待するデータでないことから、重複ACKを有線側ホストに転送することになる(figure3.4のduplicate ack)。この重複ACKが3つ以上届くと、有線側ホストは更にパケットロスが起きたと判断し、再送を行う(figure3.4のFast retransmit)。このような悪循環によって、スループットの低下が生じる。

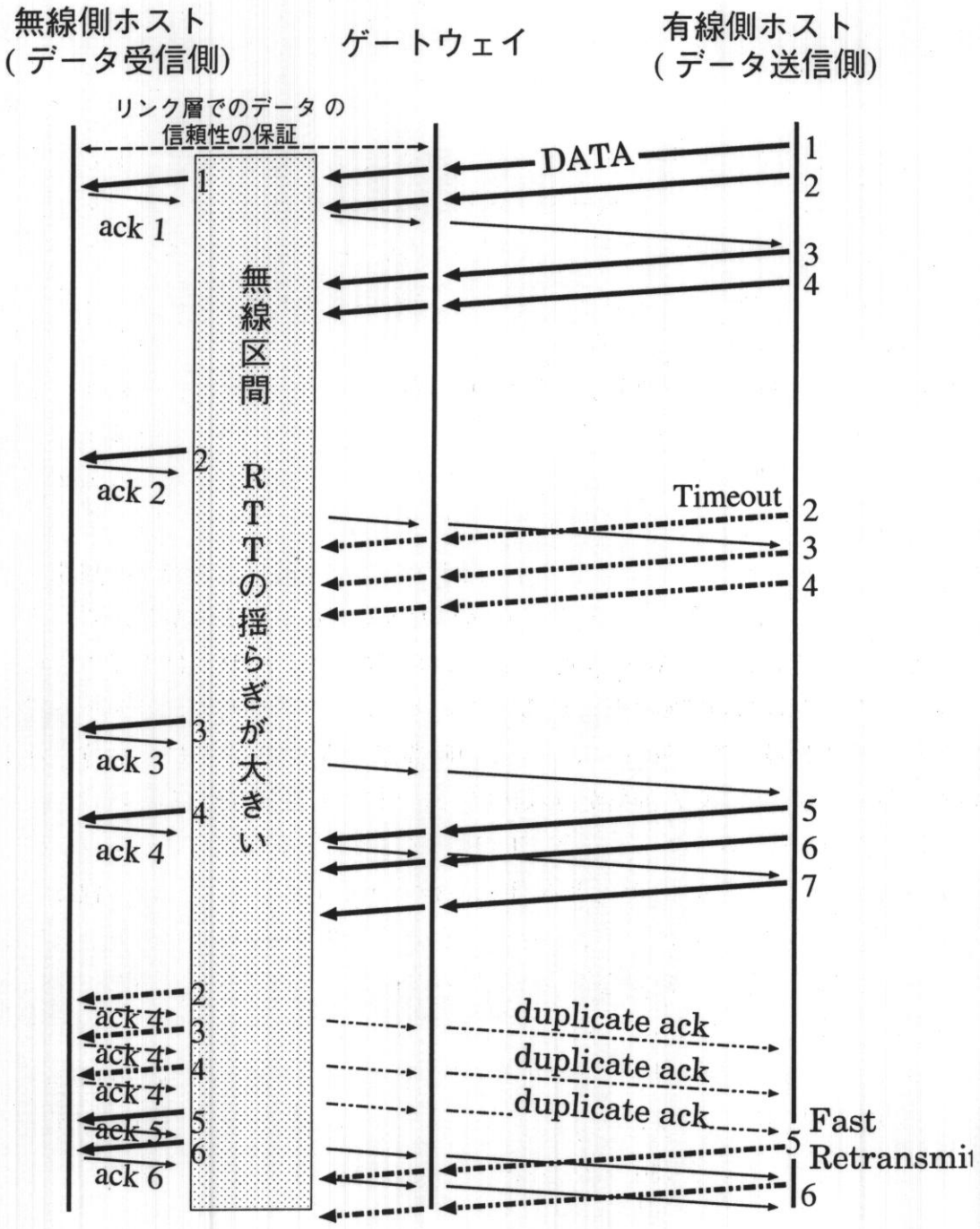


Figure 3.4 広域無線メディア利用時の TCP

## 3.3. 関連研究

前節でみたような無線通信メディアとTCPとの整合性の悪さは、これまでにもいろいろな所で指摘されている。この問題点に対して過去に研究が行われ、様々な手法が提案されている。その中でも本研究に関連する研究の幾つかを紹介する。

### 3.3.1 SACK(Seclective Acknowledgement)

通常のTCPでは累積ACK番号しか通信相手に通知できないので、送信ウィンドウ内での複数のパケットロスがあった場合に、ロスしたパケットのみの再送要求ができない。そのような状況で性能を向上させる手段としてSelective Acknowledgement(選択式確認応答)がある。この機能はTCPオプションとして定められている[16]。SACKオプションは、TCPヘッダ内の確認応答番号に加え受信側バッファに蓄積されたセグメントのシーケンス番号の情報を送信側に与えることができる。これにより、送信側はどのセグメントが紛失したかを知ることができ、そのセグメントのみを再送することで無駄な再送を省くことできる。このような機能を有したSACKオプションの導入により、高遅延、広帯域ネットワークを経由するTCPによる通信の通信効率が向上した。

しかしながら、SACKオプションを有するTCPによる通信では、送信側はどのセグメントを送るべきかを知ることができても、いつ、どれだけのセグメントを送るべきかを知ることはできないという問題点も抱えている。それはネットワーク内に存在する未処理データの正確な量を知ることができないからである。

その問題点に対しては、SACKオプションにより得られる情報を基にネットワークに流れるデータ量を厳密に見積もることにより、輻輳の発生を防ぐFACK輻輳制御アルゴリズムが提案されるなど改良が進められている。[17]

### 3.3.2 ELN(Explict Loss Notification)

無線区間でビット誤りなどの輻輳以外の原因でパケットロスが起きた場合でも、送信側ホストはそのパケットロスを輻輳が原因だとみなし輻輳回避を起動する。そこで、輻輳以外の原因でパケットロスが生じた場合、ACKにELN(Explict Loss

Notification) オプションを付加してやることで送信側ホストに通知し、その場合には輻輳回避を起動させないようにすることで解決を計るという手法である。

チェックサムエラーなどでパケットを破棄してしまうような時には、その情報を TCP での ELN オプションを付加した ACK を転送することで受信側に通知することが可能である。しかし、無線 LAN などではハンドオーバ時にはパケットそのものが失われ、受信側に届かない場合がある。このような時には ELN オプションで輻輳以外の原因でパケットロスが起きていることを通知することが困難となる。

### 3.3.3 Split Connection Approach

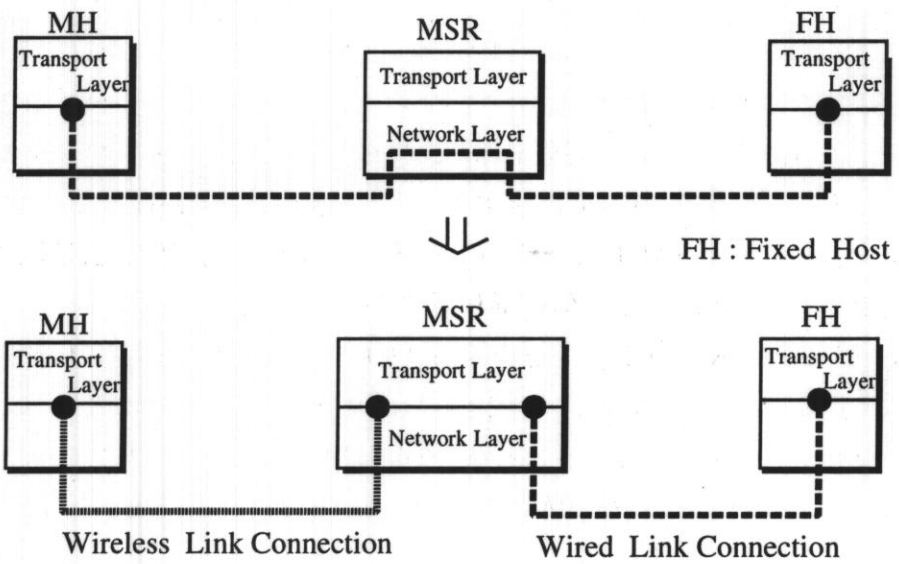


Figure 3.5 Split Connection Approach

次にコネクション分割 (Split Connection) と呼ばれる方式について述べる。figure3.5で示すように通常 TCP のコネクションは移動ホストと通信相手のホストとの 1 つのコネクションで実現される。コネクション分割方式では、無線ネットワークと有線ネットワークの移動支援ルータ (Mobile Suport Router:MSR) と呼ばれる中継地点ルータで TCP のコネクションを分割し、移動ホストと移動支

援ルータ間、移動支援ルータと通信相手ホスト間で別々のコネクションを用いる。コネクションを分割することにより、前述した無線ネットワーク側の影響を有線ネットワークと分離できる。無線側に独自のフロー制御、パケット再送機構を用いることも可能で、パフォーマンスの向上を計ることができる。

**I-TCP Protocol**

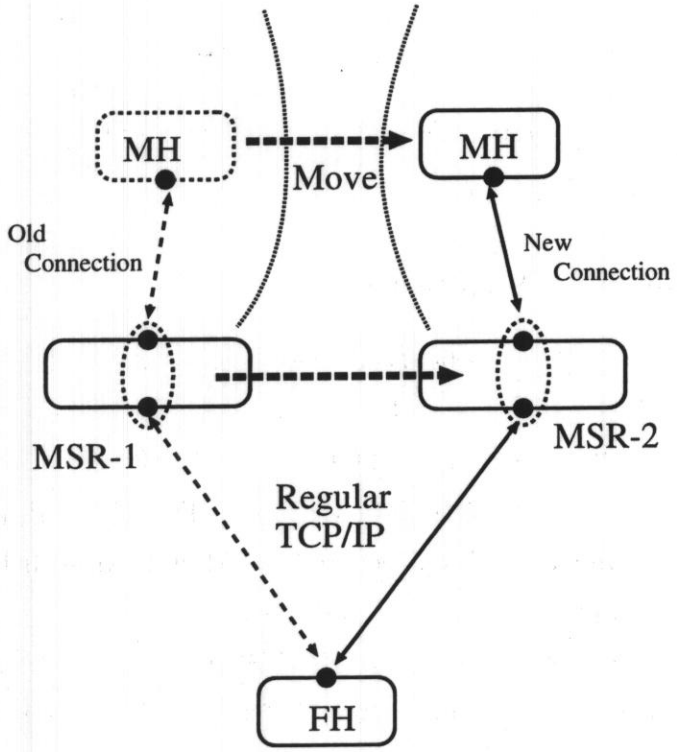


Figure 3.6 I-TCP Protocol のハンドオーバ処理

I-TCP(Indirect TCP) では移動ホストと固定ホストとの通信において MSR で TCP コネクションを移動ホストと MSR、MSR と固定ホストと 2 つに分割する [12]。コネクション分割により、移動ホストのハンドオーバや無線側でのパケットロスを隠蔽する。移動ホストは MSR で管理、登録されるが、その際の処理は前述した Mobile IP プロトコルが用いられ、I-TCP のプロトコルを実現するデーモンプロセスと協調することで移動ホストを管理する。例えば、移動ホストとイ

ンターネット上の固定ホストのプロセス間で通信が行われるときには、MSR に移動ホストとの通信用のソケット、固定ホストとの通信用のソケットの計 2 つのソケットが生成される。

I-TCP の欠点はハンドオーバ時のオーバヘッドである。移動ホストが現在のセルから次の移動先のセルに移る際に 2 つの通信を中断し、figure3.6のようにコネクション情報を MSR 間で転送しなければならない。ソケット構造体、インターネット PCB(Protocol Control Block)、TCP 制御構造体 (TCP Control Block) を MSR 間で転送する。これにはソケットバッファ等も含まれるので、バッファサイズが大きくなれば、それ分ハンドオーバの処理に時間を要する。

もう一つの欠点として、TCP のコネクションを分割してしまうので TCP の通信セマンティクスが壊れるという問題点が挙げられる。

### Mowgli Communication Architecture

Mowgli Archtecture はデジタル携帯電話等の広域無線網を介してインターネットへ接続する時の、移動ホストと MSR で使われるプロトコルから API までも含めた通信アーキテクチャである [14]。I-TCP と同じコネクション分割のアプローチである。無線ネットワーク側のプロトコルとして I-TCP が TCP/IP を用いるのに対し Mowgli Communication Architecture では MDCP(Mowgli Data Channel Protocol) と呼ばれる独自プロトコルを用いている。

Mowgli Architecture が対象としている広域無線網は GSM(Global System for Mobile Communication) である。GSM で採用されているデータリンク層である RLP(Radio Link Protocol) は強力なエラー訂正機構を提供している。Mowgli には 2 つの動作モードがあり、Default Mode ではエラーフリーの機能をデータリンク層あるいはハードウェアに依存し、Error Monitoring Mode では下位層にエラーフリーを期待できないので、チェックサム計算によりエラー訂正を行っている。

通信を行う時には移動ホストは Mowgli Socket API と呼ばれる API 群を用いて MDCS(Mowgli Data Channel Serice：MDCP を実現するモジュール) に要求を出してコネクションを生成する。この生成した無線側のコネクションに対して、コネクションの優先度、リンク再確立機能の有無、等の属性を API を通じて設定でき、またその属性値を動的に変更することも可能である。

Mowgli Architecture では移動ホストの IP アドレスは MSR の仮想インタフェース (Virtual Interface) に割当てられ、その IP アドレス宛のパケットを受信すれば、MDCP のパケットに再構成され移動ホストに転送される。

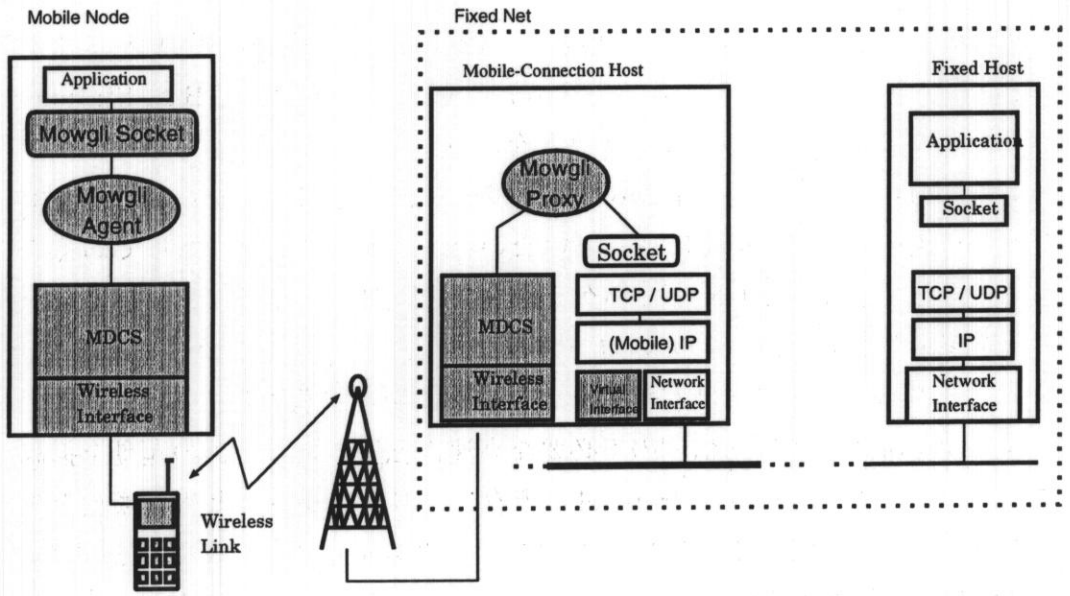


Figure 3.7 Mowgli Communication Architecture

### LWPA(Light Weight Protocol Architecture)

LWPA[15] では I-TCP、Mowgli のようにリンクの性質が明らかに変わる点にバッファを設けて 2 つの TCP コネクションに分割して通信を行う方法を採用している。また無線区間などの部分には Mowgli と同様に独自プロトコル (LWP,Light Weight Protocol) を採用して、データリンク層の特徴に合わせて効率良く通信が行うことを可能にしている。

また Mowgli のようにアプリケーションをそのプロトコルを利用するように変更するのではなく、LWP と TCP/IP とのプロトコル変換をゲートウェイで行うことによって最終的なエンド間の通信をしている。そのため、ゲートウェイの先の無線側ホストが TCP のエンドホストである必要はない。無線側ホストの先に

再び有線環境のネットワークがあり、その先にエンドホストがあるような環境においても通信は可能である。

### 3.3.4 Snoop Protocol

WaveLAN 等の無線イーサネットを用いた無線 LAN の環境での TCP パフォーマンスを向上させるプロトコルである。基本的なアイデアは基地局 (Base Station：カバーしているセル内のホストへパケットをブロードキャストする) のルーティングコードに修正を加え、Snoop Module を組込み、双方向の通過して行く全てのパケットを調べて ACK を受け取っていないパケットをキャッシュしておくことである。そして、重複 ACK 等でパケットロスを検知し、キャッシュしていたパケットを参照し無線リンク内でのローカルな再転送を行う。こうすることで、無線リンクでのパケットロスは有線ネットワーク側から隠蔽される。

Snoop Protocol の利点は送信側の不必要な高速再転送と輻輳制御の起動を抑制することである。また、Snoop Protocol は既存アプリケーションとの互換性を重視しており、I-TCP のように TCP のセマンティクスを壊すこともない。また、通常の TCP よりも 20 倍近いスループットを実現しており、ハンドオーバ時にはマルチキャストを用いた BS 間の Inteligent なパケットバッファリングを行い、パケットロスの少ないセル間の移動が可能である。

# Chapter 4

# 協調ゲートウェイの提案

本研究は広域無線通信メディア利用時の TCP の性能改善を目的とする。

前章でみたように無線環境では電波が周辺の影響を受けるために bit 誤り、瞬断、RTT の拡大などが起きる。これらの現象に対して TCP ではネットワークの輻輳が原因でパケットロスが生じていると判断し、パケットの再送、輻輳ウィンドウの減少などの輻輳回避を起動する。この輻輳以外の現象に対して輻輳回避を起動することでスループットの減少や不必要な再送といった問題が起きていた。

そこで、本研究では無線区間と有線区間を分割するゲートウェイにおいてエンドホスト間の TCP を補助することで性能改善を図る協調ゲートウェイを提案する。

この協調ゲートウェイは次の 3 つの点において優れている。

1. 無線状況の把握を行うことが可能である
2. 無線区間等のデータリンク層での信頼性保証を有効に利用できる
3. 既存のシステムの変更は必要最小限にとどめることができる

1 についてであるが、無線状況の把握を行うことは重要である。それは無線状況の悪化によって起こる現象に対し、TCP でパケットロスの判断、輻輳回避を起動させてしまうことが問題になるため、輻輳が起きているのか、無線状況が悪化しているのかを区別することが必要があるからである。しかし、現状の TCP においてパケットロスが輻輳によるものか、輻輳以外の原因によるものかの区別

をエンドホストが行うことは非常に困難である。エンドホストは経路途中にどのようなリンクを経由しているか判断する術がないからである。ところが、無線区間と有線区間を分割するゲートウェイ (以下、協調ゲートウェイ) では無線状況の把握は可能である。無線側から ACK の返送が遅れている場合には無線状況の悪化であると判断できる。この情報をエンドホスト間の TCP で利用できれば TCP の性能改善につながる。

2 についてはデータリンク層で保証しているデータの信頼性を TCP でも保証するのは、機能が重なるので無駄が多くなる。また、無線状況などによって RTT が極端に長くなった場合、ACK の到着が遅れるために TCP でタイムアウトを起こす。しかし、信頼性を保証するリンク層を利用して送信したパケットは正確に無線側ホストに届くので、無線状況の影響で ACK の返送が遅れただけである。そのような時に TCP でタイムアウト、無駄なパケットの再送が起きるという問題が生じる。このため、協調ゲートウェイまで到着したパケットで無線側ホストへ転送したデータについては、データリンク層の誤り訂正を有効に利用し、エンドホストの TCP で再送は避けることが重要である。

3 についてであるが、現在の TCP を改良して広域無線通信メディア利用時に転送効率が下がらないように、新たなものにすることは 1 つの方法として考えられる。しかし、現在の TCP で稼働している様々なサーバやクライアントを新しいものに変更することは莫大なコストがかかり困難である。よって、既存のシステムの変更する部分は最小限にとどめ、TCP の効率の改善を図ることが重要である。

以上のことから、本研究ではデータリンク層のプロトコルによって、データの信頼性を保証している区間と保証していない区間をわけるゲートウェイが通信の補助をすることによって、TCP の性能を改善する方法を提案する。

これまでに、無線部分と有線部分との間のゲートウェイで TCP コネクションを分割したり [12]、無線区間では独自プロトコルを採用し [15]、ゲートウェイから再送を行うといった手法は存在したが、このような方法の場合、有線区間と無線区間での伝送遅延、帯域の違いにより、ゲートウェイにデータをバッファリングして保存しておく必要があった。また、TCP コネクションを分割することにより、2 つの TCP コネクションの同期をとるための機構が必要であった。

本研究では、無線状況が悪化した場合、有線側の送信ホストからのデータ転送を無線状況が回復するまで停止させることにより、ゲートウェイでのバッファリングの負荷を抑える。またTCPコネクションを分割しないことで、無線区間と有線区間の同期の処理の実現が容易であるという特徴がある。詳細は次章で述べる。

なお本研究で目標とするTCPの改善で想定する環境は、figure3.4でみたような、携帯電話やPHS等の広域無線通信メディアを利用してPPPでインターネットに接続し、ファイル等を受信する状況を想定している。

# Chapter 5

# 協調ゲートウェイの設計

本章では前章でたてた方針を基に協調ゲートウェイの詳細を述べる。

本研究の協調ゲートウェイでは、4.1で述べたように、固定網にモデムを通じて継った固定ホストと、移動ホストをPPPで接続し、携帯電話やPHS等で回線交換型のデータ通信を行うことを想定している。この場合、無線基地局と移動ホストの間はPDC,PIAFSなどのプロトコルでエラーフリーが保証される。また無線基地局とモデムの間でもMNP CLASS 4やV42などのエラー訂正プロトコルが適用されるので、この間もエラーフリーが保証される。つまり、移動ホストとモデムで継った固定ホストまではデータリンク層でエラーフリーが保証されることになるので、この固定ホストを協調ゲートウェイとする。

以下では、協調ゲートウェイが実際に行う処理についてと、協調ゲートウェイが介在した場合にTCPの挙動がどのように改善されるかについて述べる。

## 5.1. 協調ゲートウェイの行う処理

協調ゲートウェイの最も重要な役割は代理ACKの作成である。協調ゲートウェイと無線側ホストの間は、データリンク層の誤り訂正プロトコルによりエラーフリーが保証されるため、協調ゲートウェイを通過して無線側ホストへ転送されるデータに関しては無線側ホストへ正常に到達する。このことを考慮して、協調ゲートウェイでは、無線区間の先にいる受信ホストの代理でACKを作成し、有

線側の送信ホストへ返送する。これにより無線状況の影響を受け、無線側ホストからの ACK の到着が遅れ、有線側ホストの TCP でタイムアウト再送が起きるのを防ぐことができる。

また協調ゲートウェイは無線状況の把握を行う。無線側ホストから ACK の返送を確認することで、無線状況の推測が可能である。無線側ホストからの ACK の情報を代理 ACK に反映させることで、送信側ホストのデータ転送を制御する。

上記の処理を行うために、協調ゲートウェイではそこを通過する TCP コネクションを監視する。その詳細を順に述べる。

### 5.1.1 TCP コネクションの監視開始

協調ゲートウェイでは、自分を通過する TCP コネクションを監視する。TCP コネクションの識別はエンドホスト 2 組の IP アドレス、ポート番号を基に行う。コネクションの監視については、通過する TCP パケットのヘッダを覗き見ることで、通信する 2 組のホストの IP アドレス (source IP address, destination IP address)、ポート番号 (source port, destination port)、シーケンス番号や ACK 番号、受信ウィンドウサイズなどを把握することができる。それらの情報をコネクションごとに管理する。

協調ゲートウェイでは代理 ACK を作成する。それはエンドホスト間での TCP コネクションが確立されてから行う。よって、実際にエンドホスト間での TCP コネクションが確立されたかどうかを判断しなければならない。そのため、協調ゲートウェイでは監視するコネクションごとに状態を保持しておき、エンドホスト間で TCP コネクションが確立されたものに対してのみ代理 ACK の処理を行う。

TCP のコネクションが確立されるには、エンドホスト間で SYN パケットが送信される。協調ゲートウェイで一方のエンドホストからの SYN を確認した場合には、新しく監視するコネクションの情報を監視コネクションリストに加え、監視するコネクションの状態を SYN_SENT 状態に遷移させる。この状態では SYN パケットの TCP ヘッダ中のの source IP address, source port から destination IP address,destination port へ、sequence number を初期シーケンス番号とする SYN が送信されたことになる (figure2.1参照)。従って、他方のエンドホストからは source IP address と destination IP address、source port と destination port を

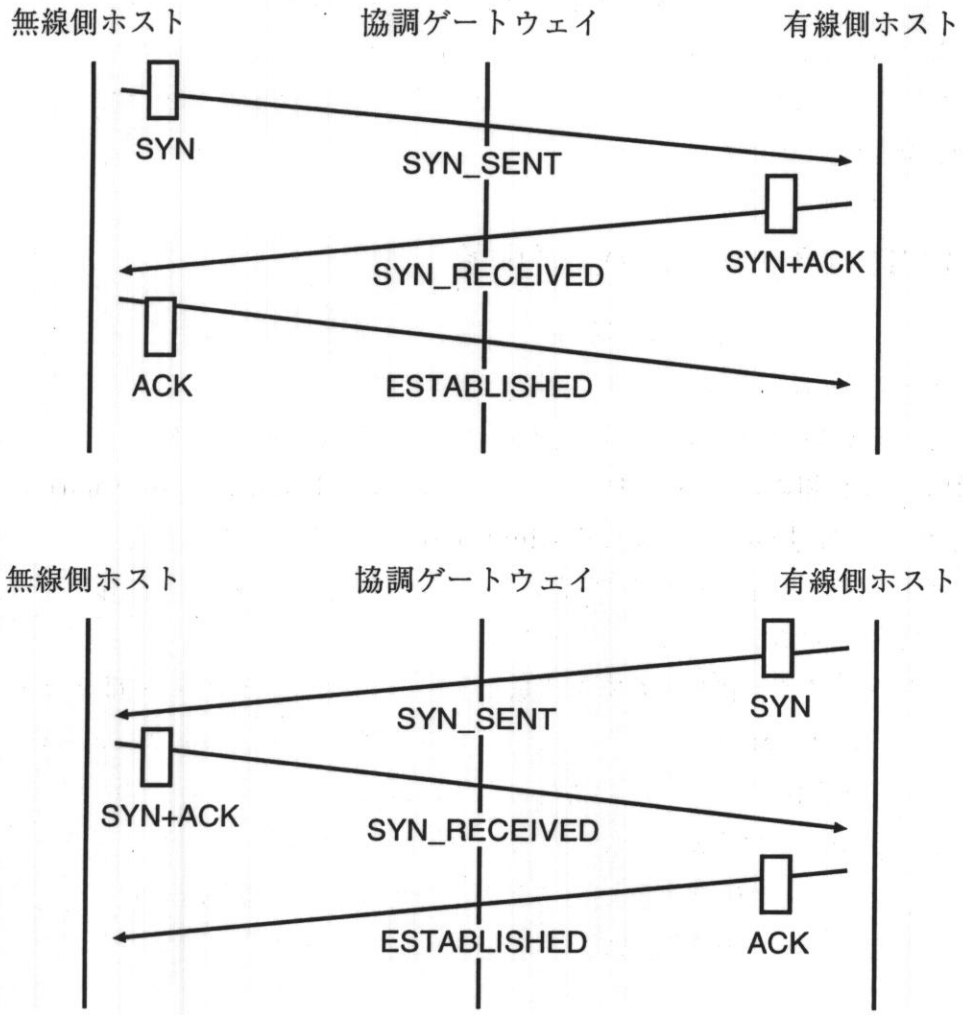


Figure 5.1 TCP コネクションの監視開始

逆にし、sequence number+1 を ACK 番号とする SYN+ACK が返送される。この SYN+ACK が通信相手のホストから返送されたことを確認した場合、監視するコネクションの状態を SYN_RECEIVED 状態に遷移させる。なお、この SYN+ACK のパケットの TCP ヘッダ中の sequence number が他方のエンドホストの初期シーケンス番号である。

次に、最初に転送された SYN の source IP address, source port から destination IP address,destination port へ他方のエンドホストの初期シーケンス番号+1 を ACK 番号とする SYN+ACK に対する ACK が返送されたことを確認した場合、監視するコネクションの状態を ESTABLISHED 状態に遷移させる。ESTABLISHED 状態のコネクションに対してデータの転送が開始されると、協調ゲートウェイでの代理 ACK の処理が開始される。

### 5.1.2 代理 ACK の作成・送信

協調ゲートウェイでは無線側ホストの代理 ACK を作成する。その際に無線側ホストの受信ウィンドウの大きさを考慮した ACK を作成する。協調ゲートウェイは、無線側ホストの受信ウィンドウの大きさを TCP コネクション確立の際に知ることができる。協調ゲートウェイではコネクションごとに常に無線側ホストの受信ウィンドウの理論値を計算する。

有線側ホストからの正常の順番通りのデータを受信した場合、そのシーケンス番号とデータ長を基に ACK 番号を代理 ACK に反映させると共に、協調ゲートウェイで保持している無線側ホストの受信ウィンドウの理論値からデータ長を引いた値を新たな理論値とする (figure5.2)。その無線側ホストの受信ウィンドウの理論値を代理 ACK のウィンドウサイズに反映させ、有線側ホストへ転送する。

また無線側ホストから ACK が返ってきた場合、保持していた無線側ホストの ACK 番号よりも、無線側から返ってきた ACK の ACK 番号が進んでいれば、進んだ分だけのデータが処理されたと判断し、協調ゲートウェイで保持している無線側ホストの受信ウィンドウの理論値を増加させる。協調ゲートウェイでは代理 ACK を送信しているので、無線側ホストから届いた ACK は破棄し、有線側ホストに転送しない。

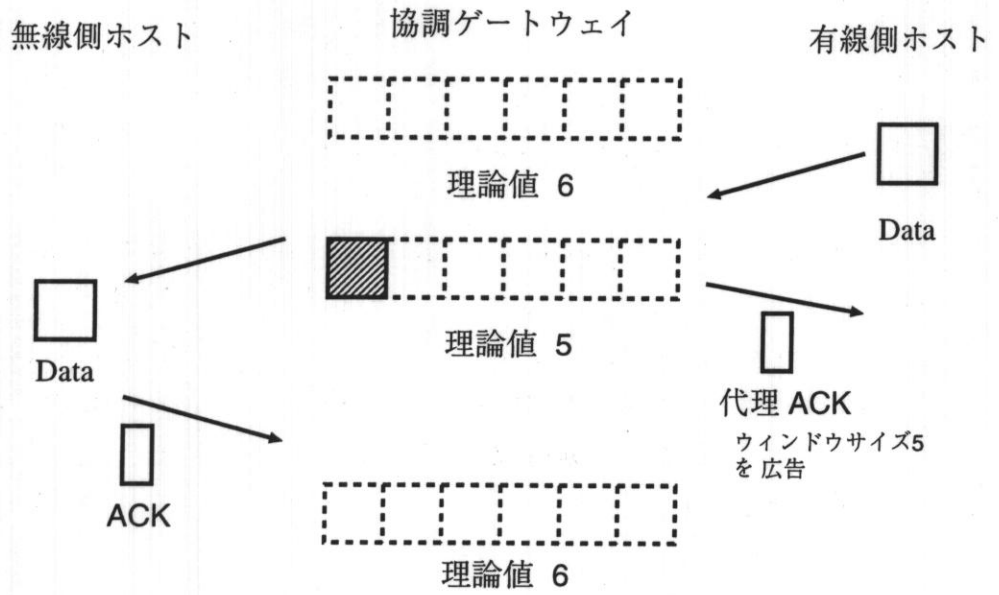


Figure 5.2 協調ゲートウェイで保持する無線側ホストの受信ウィンドウサイズの理論値

無線状況が良好であれば問題はないが、無線状況の悪化時には無線側ホストからの ACK の到着が遅れだす。すると有線側ホストからのデータが転送されてくる度に協調ゲートウェイで保持している無線側ホストの受信ウィンドウの理論値は減少していき、ついには 0 になる。0 ウィンドウを広告する ACK が返されると有線側ホストはそれ以上のデータ転送を行うことができなくなる。これにより無線状況が悪化した場合には有線側ホストのデータ転送を停止させることが可能となる。

TCP の仕様では 0 ウィンドウを広告されてデータ転送ができないホストは、タイマーを設定し時間内にウィンドウサイズが更新された ACK が到着しない場合、ウィンドウサイズが更新されたかどうかを確認するため、1 バイト分のデータを転送する。そのウィンドウ更新確認のデータが協調ゲートウェイに送られてきた場合、ACK 番号を更新することなく、再び 0 ウィンドウを広告する代理 ACK を送信する必要がある。

協調ゲートウェイで保持している無線側ホストの受信ウィンドウの理論値が 0

で、無線側ホストからACKが到着し、理論値が0から増加した場合には、協調ゲートウェイは更新されたウィンドウサイズを広告する代理ACKを作成し、有線側ホストに送信する(figure5.3)。

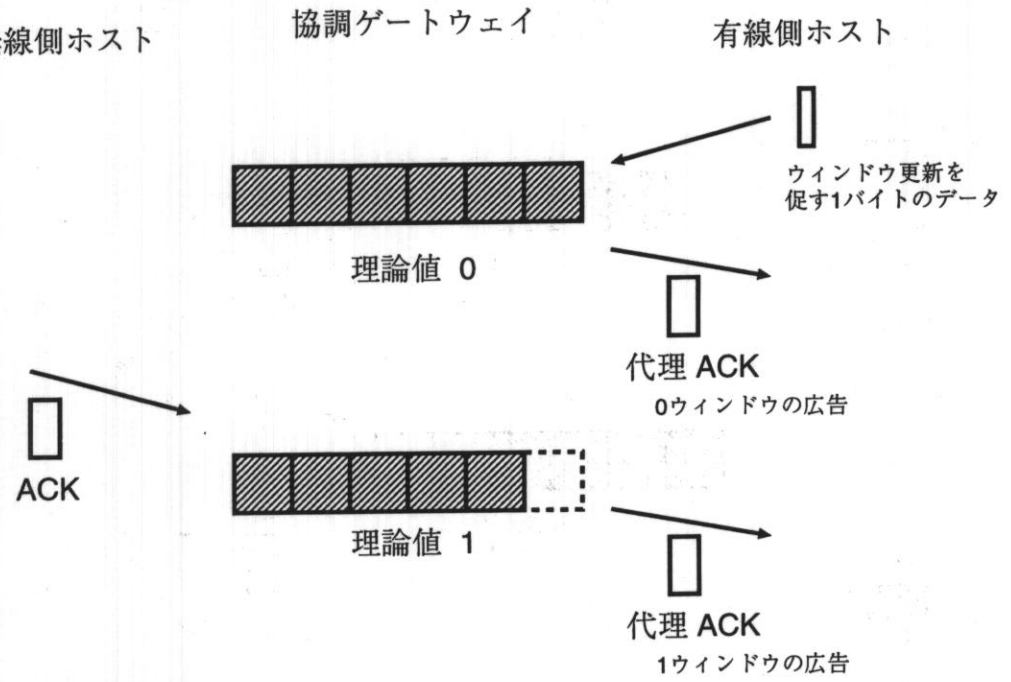


Figure 5.3 協調ゲートウェイで保持する無線側ホストの受信ウィンドウサイズの理論値が0から更新された場合の処理

### 5.1.3 順番誤りで到着したデータパケットに対する処理

協調ゲートウェイには有線側ホストからのデータは順番通りに到着するとは限らない。途中のルータ等の輻輳によりパケットがロスされることがありうる。このような順番誤りで到着したパケットに対しての代理ACKは処理が異なる。

ACK番号、受信ウィンドウサイズ共に更新を行わずに代理ACKを作成し、有線側ホストに転送する。この代理ACKが重複ACKとなる。有線側ホストにはタイムアウト再送、又は重複ACKによる再送を促す。

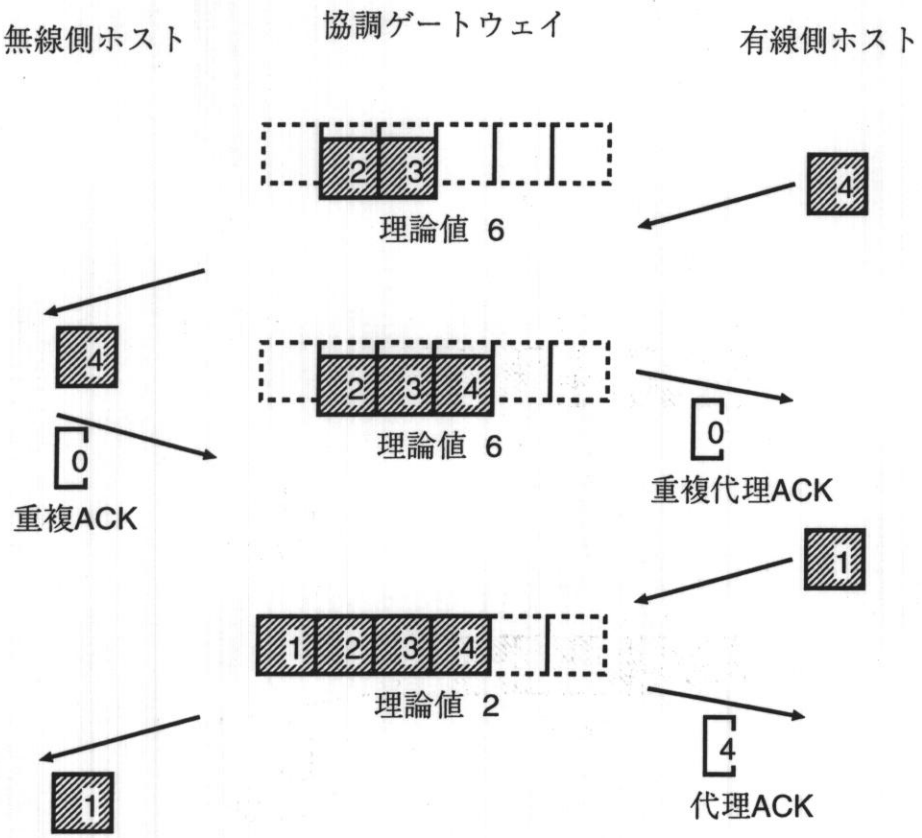


Figure 5.4 順番誤りのパケットに対する協調ゲートウェイの処理

順番誤りのパケットに関してはそのまま無線側ホストへ転送する。その際、協調ゲートウェイでは転送した順番誤りのデータの最初と最後のシーケンス番号を保持しておく。次に抜けた部分のデータが再送され、協調ゲートウェイで保持していた順番誤りのデータと継った場合、代理 ACK の ACK 番号は保持しておいた最も大きいシーケンス番号を代理 ACK の ACK 番号とする。またウィンドウサイズは継ったデータ分を減少させる。

figure5.4では 1 番のパケットが抜け、2 番から 4 番までのパケットが転送されてきた状況を想定している。順番誤りで到着したパケットに対しては代理重複 ACK を有線側ホストへ転送、データパケットはそのまま無線側ホストへ転送する。その際、無線側ホストへ転送したデータ (2 から 4 まで) のシーケンス番号は記憶し

ておく。タイムアウト、重複ACKの受信による再送等で抜けていた部分の1番のパケットが転送されると、送信側ホストで1から4までのデータがTCPなどの上位層が渡されることとなる。

よって、協調ゲートウェイでの代理ACKでは4番のパケットまで受信し、空き受信ウィンドウサイズは2としている。その後の協調ゲートウェイで保持する無線側ホストの受信ウィンドウサイズの理論値は、無線側ホストからのACKにより再び計算される。

### 5.1.4 TCPコネクションの監視終了

エンドホスト間でTCPコネクションが終了すると、そのコネクションを協調ゲートウェイの監視コネクションリストから外す。通常のTCPでのコネクションの終了はエンドホストがFINを送信し、それに対するACKが他方のエンドホストから返されるという処理が、両エンドホストで行われると完了する。協調ゲートウェイで片方のエンドホストからのFINを確認した場合、そのコネクションのどちらのホストからFINが送られたのかをチェックしておく。そのFINに対するACK、他方からのFIN、それに対するACKを確認した段階で、コネクションを監視リストから外し、協調ゲートウェイでの監視処理が終了する。

また、TCPのコネクションの終了にRSTがある。これは通信の強制終了である。RSTが送信されたTCPコネクションは通信の途中であっても強制終了させられる。よって、協調ゲートウェイで監視しているコネクションにおいてRSTパケットを検出した場合、そのパケットを転送すると共に、監視コネクションリストからそのコネクションを外す処理を行う。

### 5.1.5 再送パケット処理

協調ゲートウェイでは無線側ホストの代理ACKを作成、送信するので、無線状況の影響を受けることによる有線側ホストでのタイムアウト再送は起こらない。しかし、協調ゲートウェイで送信した代理ACKが有線側ホストに到達する途中の経路で失われることによって、タイムアウト再送が起きることは可能性がある。この場合、有線側ホストから再送パケットが送信される。しかし、以前送信され

無線側ホストに転送したパケットと同じものでがデータリンク層での誤り訂正プロトコルにより正常に無線側ホストに到達する。よって、再送パケットを無線側ホストへ転送すると、このパケットは破棄され、重複 ACK を引き起こし、データの転送効率を下げてしまう。以上のことから、協調ゲートウェイで確認される有線側ホストからの再送データに関しては、協調ゲートウェイで破棄し、無線側ホストに転送しない。

また、無線状況が悪化している時に、無線側ホストがデータを転送しようとした場合、そのパケットに対する ACK の到着が遅れ、無線側ホストで再送が起きる。協調ゲートウェイでは TCP コネクションを絶えず監視しているので、無線側ホストからの再送パケットも検知することができる。そして、以前無線側ホストから送信されてきたそのパケットに対して、有線側から既に ACK が返送されているかどうかも判断できる。既に有線側ホストから ACK が返送されているデータが無線側ホストから再び送られてきた場合には、協調ゲートウェイではそのパケットを破棄して無駄なデータを転送することは避ける。

## 5.2. 協調ゲートウェイの介在した TCP の具体例

figure5.5は協調ゲートウェイが介在した場合の TCP の挙動を示したものである。

協調ゲートウェイから無線側ホストになりすました代理 ACK が送信されている。図中に書かれた ack1(5) などの括弧の中の数字は、無線側ホストの受信ウィンドウサイズを示す。受信ウィンドウサイズは TCP コネクション確立の際、SYN をやりとりする時に協調ゲートウェイでパケットを覗き見ることで把握することができる。図中の TCP コネクションでは、無線側ホストの受信ウィンドウは 6 セグメント分である。従って、最初の 1 セグメントが送信された際に、協調ゲートウェイでは、無線側ホストの受信ウィンドウ 6 のうち、1 セグメント分データが占めたという想定をし、代理 ACK の受信ウィンドウサイズの欄に 5 セグメント分受信可能であることを広告する。

無線側ホストから実際の ACK が協調ゲートウェイに返ってきた場合は、改めて受信可能ウィンドウサイズを計算し、次に有線側ホストからきたデータパケットに対する代理 ACK に反映させる。figure5.5の代理 ack3 において広告受信ウィ

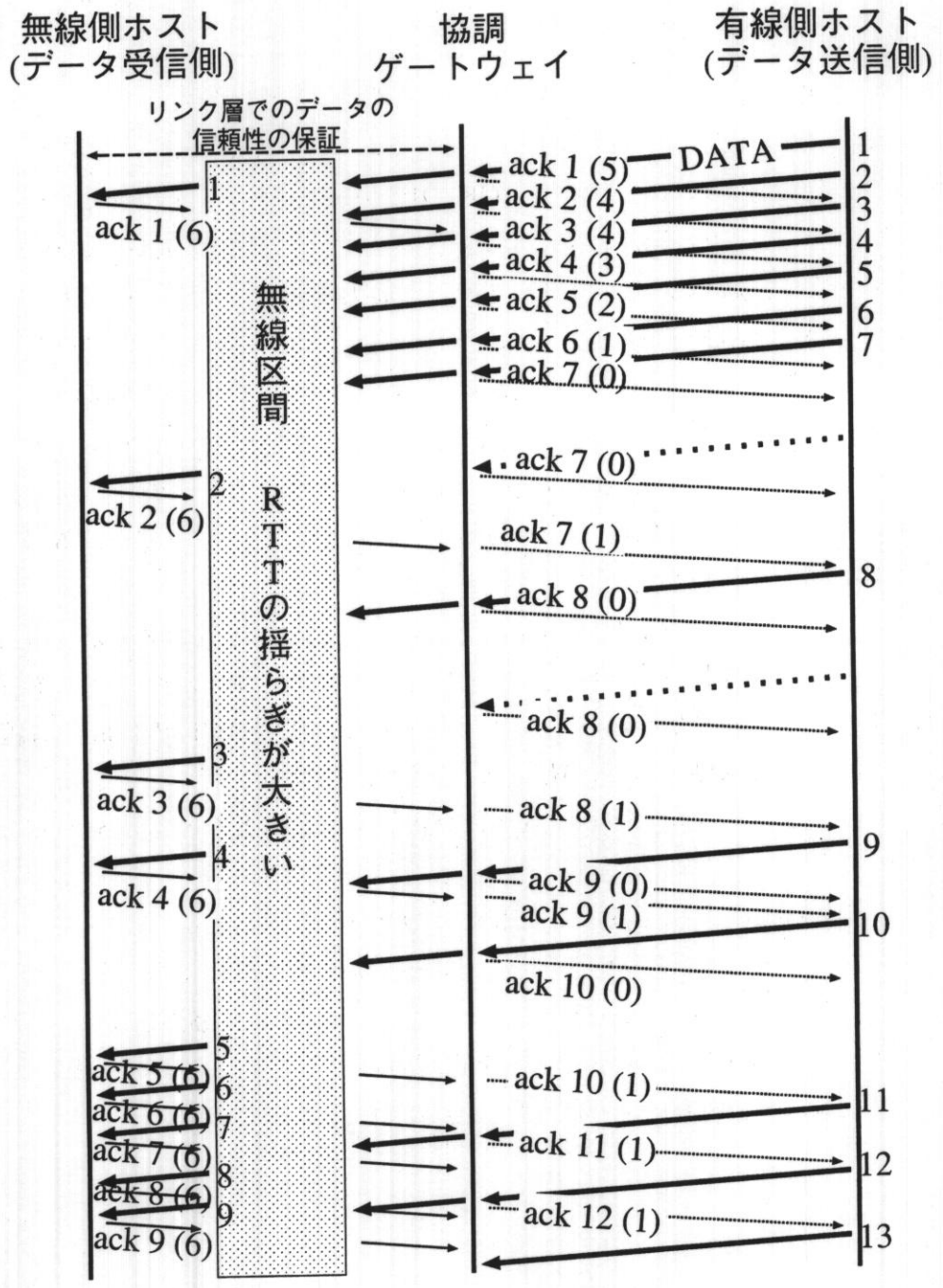


Figure 5.5 ゲートウェイ協調による TCP

ンドウがack2と同じ値の4であるのは、無線側ホストからのack1を受信したことで、受信ウィンドウサイズが1セグメント分の空きができたことを確認したからである。

無線状況の悪化により、2つめのセグメントの到着が遅れ、無線側ホストからのack2の到着が遅れる間にも、協調ゲートウェイの代理ACKにより、有線側ホストのデータ転送は進み、やがて代理ACKの広告受信ウィンドウが0になり、有線側ホストはデータの転送ができなくなる。このように無線状況が悪い時は、有線側のホストには無線側ホストのデータの処理能力が低く、転送されたデータの処理が遅れていると判断させることで、通常のTCPのようにタイムアウトの発生、それに伴う無駄なパケットの再送が起きることを防ぐことができる。

TCPでは0ウィンドウが広告されると、送信側は、受信側の受信可能ウィンドウサイズが更新されたかを確認するために1バイト転送する。そのパケットに対しては再び0ウィンドウの代理ACKを送信することで対処する(figure5.5における2つの点線部)。

0ウィンドウを広告している間に、協調ゲートウェイへ無線側ホストからackが返送され、無線側ホストの受信可能ウィンドウサイズが更新されたことを確認した場合には、その空き受信ウィンドウサイズをもとに代理ACKの作成、有線側ホストへ送信し、有線側ホストに続きのデータ転送を促す。

このようにして無線状況の悪い時にも無駄な再送をすることなくデータの転送を行うことができる。

figure5.5とfigure3.4は同じ無線状況のもとでのTCPの挙動を示している。これら2つを比較してみると、通常のTCPでは、無駄な再送が起きることで、正味7パケット分しか送信されていないところを、ゲートウェイ協調によるTCPは無駄な再送が起こらずに、13パケットを送信可能としている。このことからゲートウェイ協調によりTCPが性能を向上させているといえる。

# Chapter 6

# 協調ゲートウェイの実装

本章では5章の設計に基づいて協調ゲートウェイの実装について述べる。実装はBSD/OS 3.1上でiij-pppのコードに協調ゲートウェイのコードを追加するかたちで行った。

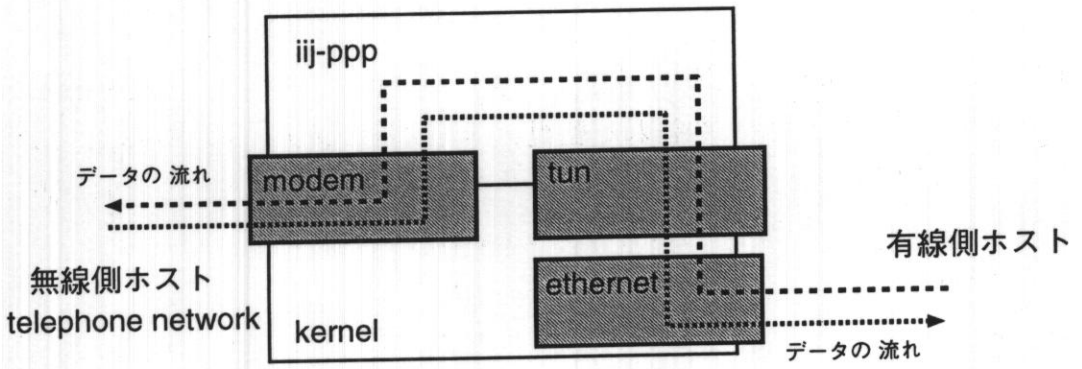


Figure 6.1 iij-pppの構造

figure6.1はiij-pppの構造を示した図である。iij-pppはユーザ空間で動くアプリケーションである。有線側ホストからのデータはイーサネット等からカーネル、トンネルデバイスを通してiij-pppのプロセスに渡される。そのデータをモデムに渡すことで公衆電話網を通して無線側ホストへ転送される。また、無線側ホストからのデータは逆に、モデムを通してiij-pppのプロセスに渡される。これを

トンネルデバイスに渡すことでカーネル、イーサネット等を経由して有線側ホストに転送される。

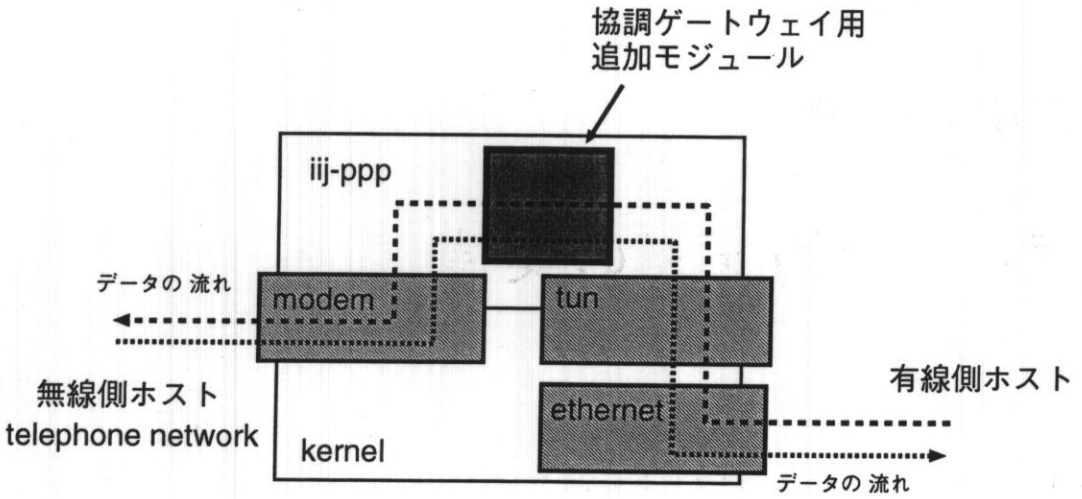


Figure 6.2 iij-ppp へ協調ゲートウェイモジュールの追加

本研究ではこの iij-ppp に協調ゲートウェイモジュールを追加した。figure6.2のように通過する TCP パケットの流れを監視、制御するモジュールである。このモジュールでは TCP コネクションごとにコネクション管理ブロックと、順番誤りでパケットが転送されてきた時に管理する順番誤りデータ管理ブロックというデータ構造 (figure6.3) をもつ。

コネクション管理ブロックでは TCP コネクション識別子とコネクションに関するデータを保持する。コネクションの識別にはエンドホスト 2 組の IP アドレスとポート番号の組が使われる。コネクションに関するデータとしてはコネクションの状態、双方向のシーケンス番号、ACK 番号、代理 ACK 番号、ウィンドウサイズの理論値などのコネクション管理に必要なデータを保持する。

順番誤りデータ管理ブロックでは順番誤りで転送されてきたデータのシーケンス番号とデータ長からデータストリームのどの部分が誤って転送されてきたのかという情報を保持する。

協調ゲートウェイ用モジュールが行う処理として、次のものが挙げられる。

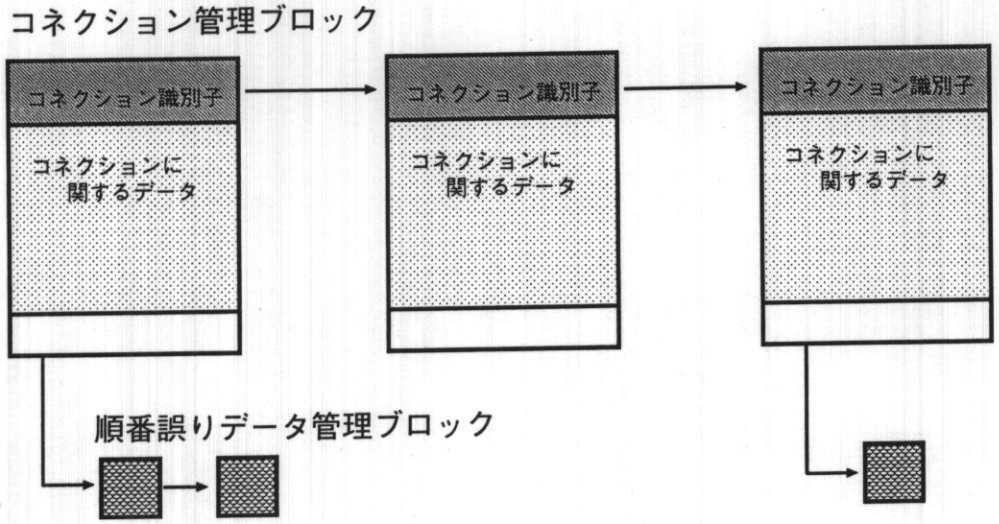


Figure 6.3 協調ゲートウェイモジュールで使用するデータ構造

- コネクション管理ブロックの作成
- トンネルデバイスから渡されるデータの処理
- モデムから渡されるデータの処理
- コネクション管理ブロックの消去

## 6.1. コネクション管理ブロックの作成

コネクション管理ブロックの作成に関してはSYNを検知し、コネクション管理ブロックリストにそのコネクションが見つからない場合、新規コネクションと判断し、IPアドレスとポート番号の組から新たな識別子を有するコネクション管理ブロックを作成し、割り当てる。その際にモデムからの渡されたSYNかトンネルデバイスから渡されたSYNかを基に、有線側ホストからのSYNか、無線側ホストからのSYNかを区別し、どちらのIPアドレスが無線側ホストにあたるかを把握する。そのSYNに対するSYN+ACK、SYN+ACKに対するACKを確認すると、そのTCPコネクションに対する通信の補助が開始される (5.1.1参照)。

## 6.2. トンネルデバイスから渡されるデータの処理

トンネルデバイスから渡されるデータは有線側ホストからのデータ、ACKである。このデータやACKに関する処理は次の通りである。

- 有線側ホストからのデータを無線側ホストへ転送(モデムへの書き込み)、又は破棄
- 有線側ホストからのデータに対する代理ACKの作成・送信(トンネルデバイスへの書き込み)
- コネクション管理ブロックで保持しているデータの更新

有線側ホストからのデータが渡された場合、そのデータに対する代理ACKを作成する。ただし、既に無線側ホストへ転送したデータが到着した場合にはそのパケットは破棄し、代理ACKも作成しない。

データが順番通りの場合で、既に順番誤りで転送したデータがない場合には、ACK番号をデータの最後までの番号とし、受信可能ウィンドウサイズの理論値からデータの長さ分減じた値をウィンドウサイズとする代理ACKを作成する。既に順番誤りで転送したデータがある場合には、このデータが抜けた部分を埋めるデータかどうかのチェックをし、データが抜けた部分を埋める場合、連続するデータの最後の番号をACK番号とし、受信可能ウィンドウサイズの理論値から連続したデータの長さ分減じた値をウィンドウサイズとする代理ACKを作成する。(5.1.3参照)

順番誤りで到着のデータの場合、ACK番号は順番通りで受信しているデータの最後の番号とした代理ACKを作成する。それまでに順番誤りで転送したデータがない場合には、順番誤りデータ管理ブロックを作成し、パケットのシーケンス番号とデータ長からデータストリームのどの部分が順番誤りで転送されてきたのかという情報を記録する。

既に順番誤りで転送したデータがある場合には、保持しているデータの順番と連続するものかどうかのチェックを行う。連続するデータの場合、順番誤りデータ管理ブロックを更新する。連続しないデータの場合、新たな順番誤りデータ管

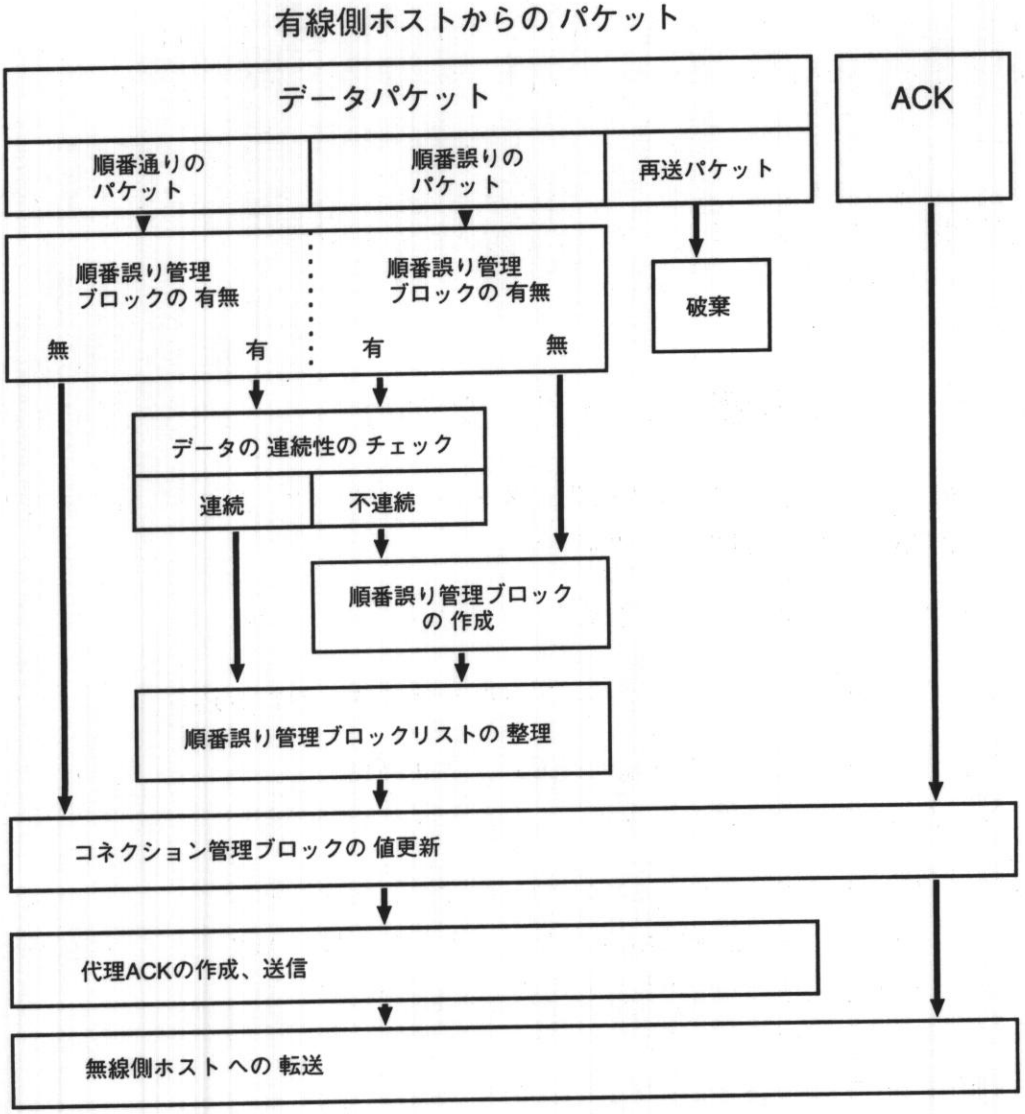


Figure 6.4 トンネルデバイスからのパケットの処理の流れ

理ブロックを作成し、順番誤りデータ管理ブロックリストでデータストリームの順番に当てはまる部分へつなぐ。

次に、コネクション管理ブロック中の有線側ホストのシーケンス番号、代理ACKのACK番号、ウィンドウサイズの理論値などを更新する。

作成した代理ACKはトンネルデバイスに書き込むことで、有線側ホストへ転送される。また、有線側からのパケットは再送データでなければ無線側ホストに転送する。

有線側ホストからのACKが渡された場合には、コネクション管理ブロック中の有線側ホストのACK番号を更新し、無線側ホストへ転送する。

これら一連のデータ処理の流れはfigure6.4の通りである。

## 6.3. モデムから渡されるデータ処理

モデムから渡されるデータは無線側ホストからのデータ、ACKである。このデータやACKに関する処理は次の通りである。

- コネクション管理ブロックで保持しているデータの更新
- データを有線側ホストへ転送(トンネルデバイスへの書き込み)、又は破棄
- 受信ウィンドウサイズの理論値が0からの更新時のみ、代理ACKの作成・送信(トンネルデバイスへの書き込み)

無線側ホストからのデータやACKが渡された場合、まず、コネクション管理ブロックで保持している無線側ホストのシーケンス番号、ACK番号、受信ウィンドウサイズの理論値などの値を最新のものに更新する。

コネクション管理ブロックで保持している無線側ホストの受信ウィンドウサイズの理論値が0で、無線側ホストからのACKにより更新された場合、そのことを有線側ホストへ広告するための代理ACKを作成、転送することで、通信の再開を図る。受信ウィンドウサイズの理論値が0でない場合は、通信を止めているわけではないので、代理ACKを作成する必要はない。また、協調ゲートウェイでは代理ACKを返送しているので、無線側ホストからのパケットがACKのみの場合、有線側ホストへ転送せずに破棄する。

無線側ホストから無線状況の影響で、再送データパケットが転送されてきた場合、そのパケットに対する有線側ホストからのACKを既に確認しているかどうかをチェックする。もし、既にACKを確認しているようなデータパケットである場合、そのデータは破棄して有線側ホストへ転送はしない。再送データパケットでない通常のデータパケットや有線側ホストからのACKがまだ転送されてきていない再送データパケットは有線側ホストへ転送される。この際、実際のパケットのTCPヘッダ中のACK番号と、代理ACKで転送したACK番号とずれが生じている可能性がある。その時はACK番号を書き換え、チェックサムを計算し直し転送する。

これら一連のデータ処理の流れはfigure6.5の通りである。

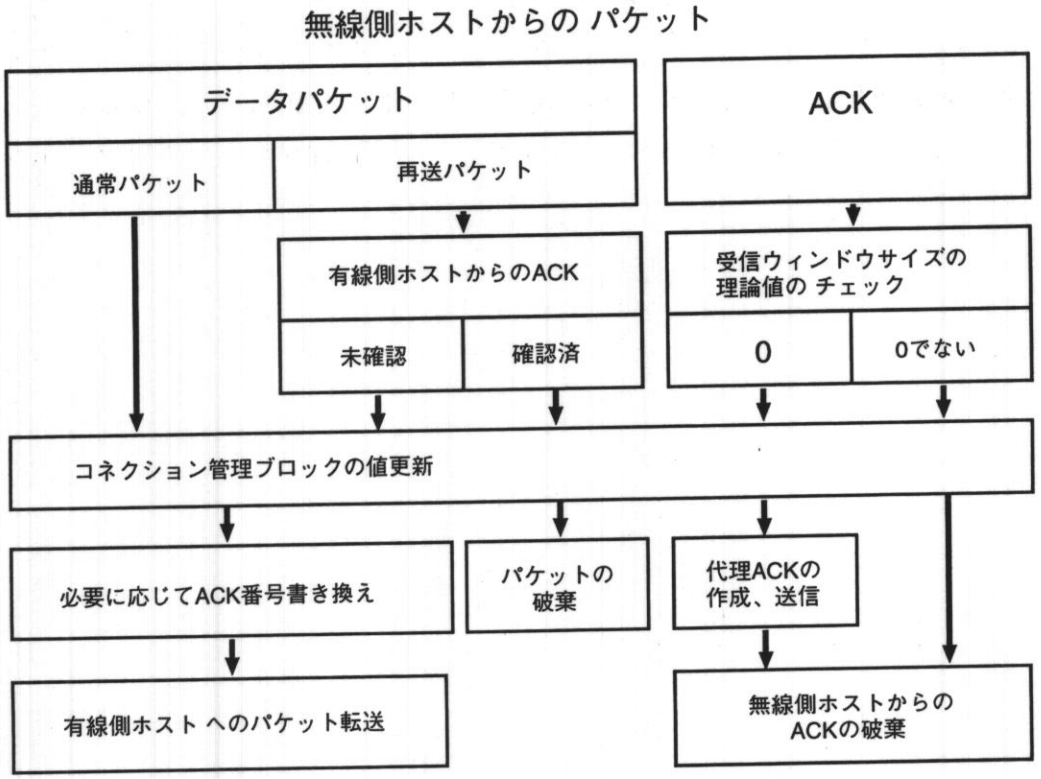


Figure 6.5 モデムからのパケットの処理の流れ

## 6.4. コネクション管理ブロックの消去

コネクション管理ブロックの消去は、両エンドホストからのFINのやりとりが行われるのを確認した後、又はRSTがエンドホストから転送されてきた場合に行う。

有線側ホストから先にFINが転送されてきた場合、協調ゲートウェイでそのFINに対する代理ACKを作成、送信する(ハーフクローズ)。後に、無線側ホストからFINが転送されてきた場合、有線側ホストへ転送し、そのFINに対するACKを待つ。無事、ACKを確認したら、そのコネクションに対するコネクション管理ブロックをリストから外し消去する。

無線側ホストから先にFINが転送されてきた場合、そのFINを有線側ホストへ転送し、そのFINに対するACKが転送されてくるのを待つ。後に、有線側ホストからFINが転送されてきたら、協調ゲートウェイでそのFINに対する代理ACKを作成、送信し、その時点で無線側ホストのFINに対するACKを確認済みであれば、そのコネクションに対するコネクション管理ブロックをリストから外し消去する。

# Chapter 7

# 協調ゲートウェイの評価

本章では協調ゲートウェイの性能評価を行う。性能評価にあたって3つの実験、測定を行った。

## 7.1. 広域無線メディアを利用した場合のRTTの揺らぎの測定

最初に広域無線通信メディアを利用した場合にデータリンク層での誤り訂正により、どの程度RTTの揺らぎが生じるのかを調べるため、実際に携帯電話を利用しPPP接続を行い、無線状況の変化の大きな地域を移動しながらPPPサーバまでのpingによるRTTの経時変化を調べた。(figure7.1)

figure7.2はその結果を示したものである。横軸はICMP sequence numberで移動ホストが送信したpingパケットの順番で、縦軸はRTT(ms)である。通常状態ではRTTが500ms程度であるが、無線状況の悪い場所へ入ると最悪の場合で8秒弱とRTTが大きくなっていることがわかる。これは3章でも述べたように携帯電話でのデータ通信ではデータリンク層でARQによる誤り訂正を行っているため、無線状況が悪くなるとデータリンク層での再送が頻発し、RTTが長くなるためである。

またRTTが大きくなる場合、その前後のパケットのRTTも影響を受けて大きくなるという特徴がある。

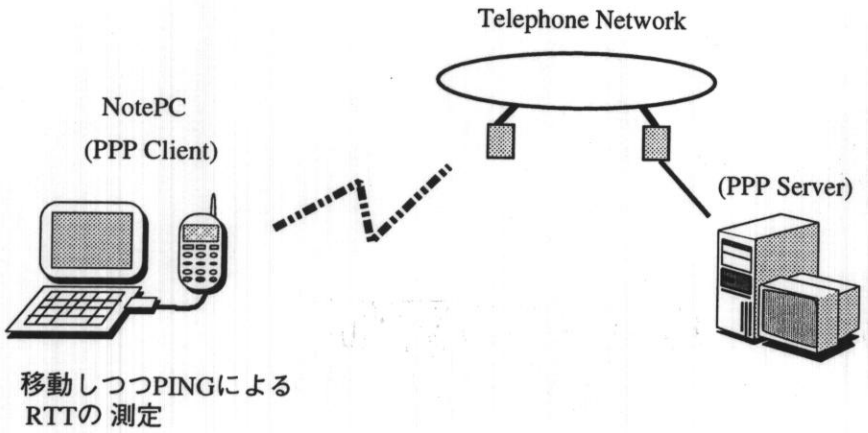


Figure 7.1 広域無線メディアを利用した場合の RTT の揺らぎの測定

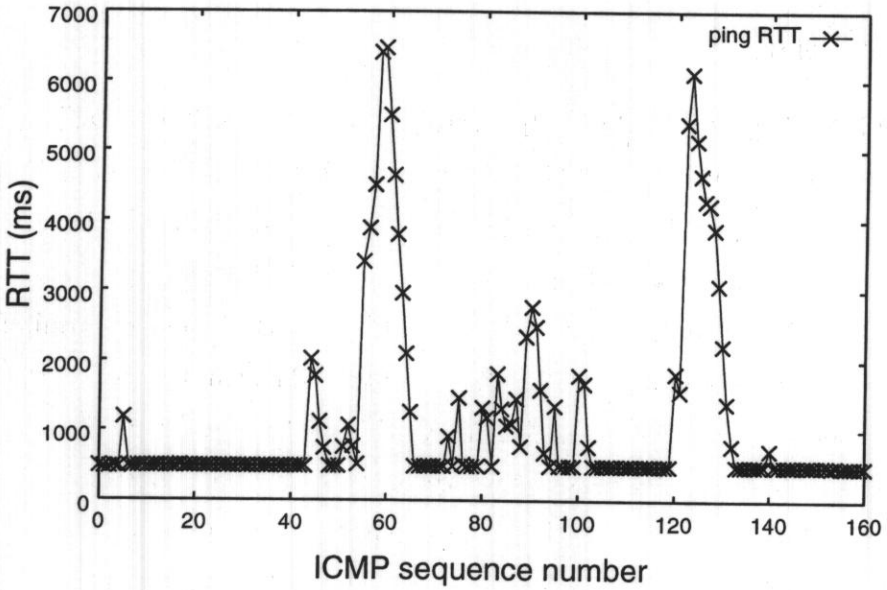


Figure 7.2 携帯電話利用時の RTT の揺らぎ

## 7.2. 無線シミュレータを用いた環境での測定

この実験では無線シミュレータを用いて、無線状況の変化をエミュレートする環境において、協調ゲートウェイの効果を測定した。

測定環境は figure7.3のようになっている。有線側ホストと協調ゲートウェイの間はイーサネットで接続され、協調ゲートウェイと無線側ホストの間は無線シミュレータを介して、無線状況をエミュレートした。協調ゲートウェイと無線側ホストの間の伝送速度は 9.6kbps とし、通常状態 (無線状況が良好な状態) では通過するパケットをそのまま通す。しかし、無線状況の悪化をエミュレートするために、定期的に一定期間、通過するパケットに対して遅延を挿入するようにした。挿入する遅延時間は 2 ～ 5 秒間とした。

**測定環境**

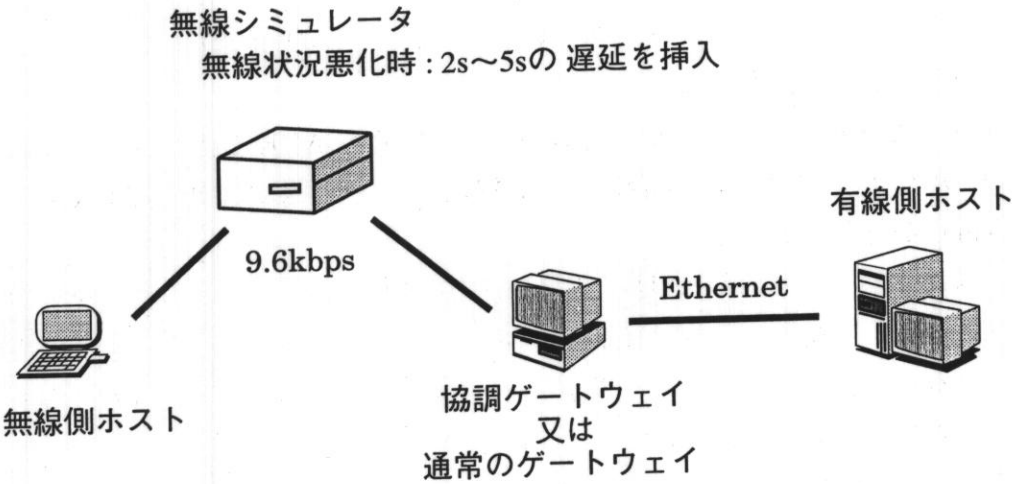


Figure 7.3 無線シミュレータを用いた環境での測定

測定方法としては、有線側ホストから無線側ホストへの 20K バイトのファイル転送を行った。その際の協調ゲートウェイと通常のゲートウェイでの TCP の動作比較をした。

figure7.4から figure7.7までは、無線側ホストで計測したパケットのシーケンス

番号と ACK 番号のタイムラインである。パケットのデータサイズは 1460 バイトである。色の付いた部分は無線状況悪化期間、つまり遅延の挿入されている期間である。設定 1 から設定 4 までは、それぞれほぼ同じ無線状況での通常ゲートウェイと協調ゲートウェイでの TCP の動作の比較である。

設定 1、設定 2 では通常期間 5 秒、無線状況悪化期間 25 秒周期に設定した。設定 3、設定 4 では通常期間 5 秒、無線状況悪化期間 55 秒周期に設定した。設定 1、設定 3 ではデータ転送初期の段階で無線状況の悪化、設定 2、設定 4 ではデータ転送の中盤で無線状況の悪化が始まっている。

どの設定でも通常のゲートウェイの場合、無線状況の悪化に伴い、有線側ホストからのデータパケットに対する ACK を返送しているが、その ACK が有線側ホストへ返るのが遅れ、有線側ホストでタイムアウトを起こす。タイムアウトを起こした有線側ホストはデータを再送する。そこへ遅れて返送されてきた以前のパケットに対する ACK を、今再送したパケットに対する ACK とみなし、続きのパケットを送信する。そのため 3 章でも触れたように、既に受信し ACK も返送したデータパケットが再送されてきている。それも一旦再送されてきたパケット以降のパケットも再送される。

これらの再送パケットに対しては受信側ホストで重複 ACK を返送する。この重複 ACK が 3 つ以上発生した場合には、有線側ホストの TCP で重複 ACK の受信による再送が行われる。

一方、協調ゲートウェイを利用した場合、無線側ホストに 1 度受信したデータが送信されることはない。これは協調ゲートウェイで代理 ACK を有線側ホストに返しているため、タイムアウト再送が起きないからである。無線状況が悪化して ACK の到着が遅れてくる場合でも、5 章で設計したように協調ゲートウェイが無線側ホストの受信ウィンドウを考慮した代理 ACK を作成し、有線側ホストのフローをコントロールしているため有効なデータパケットのみを転送させることが可能である。

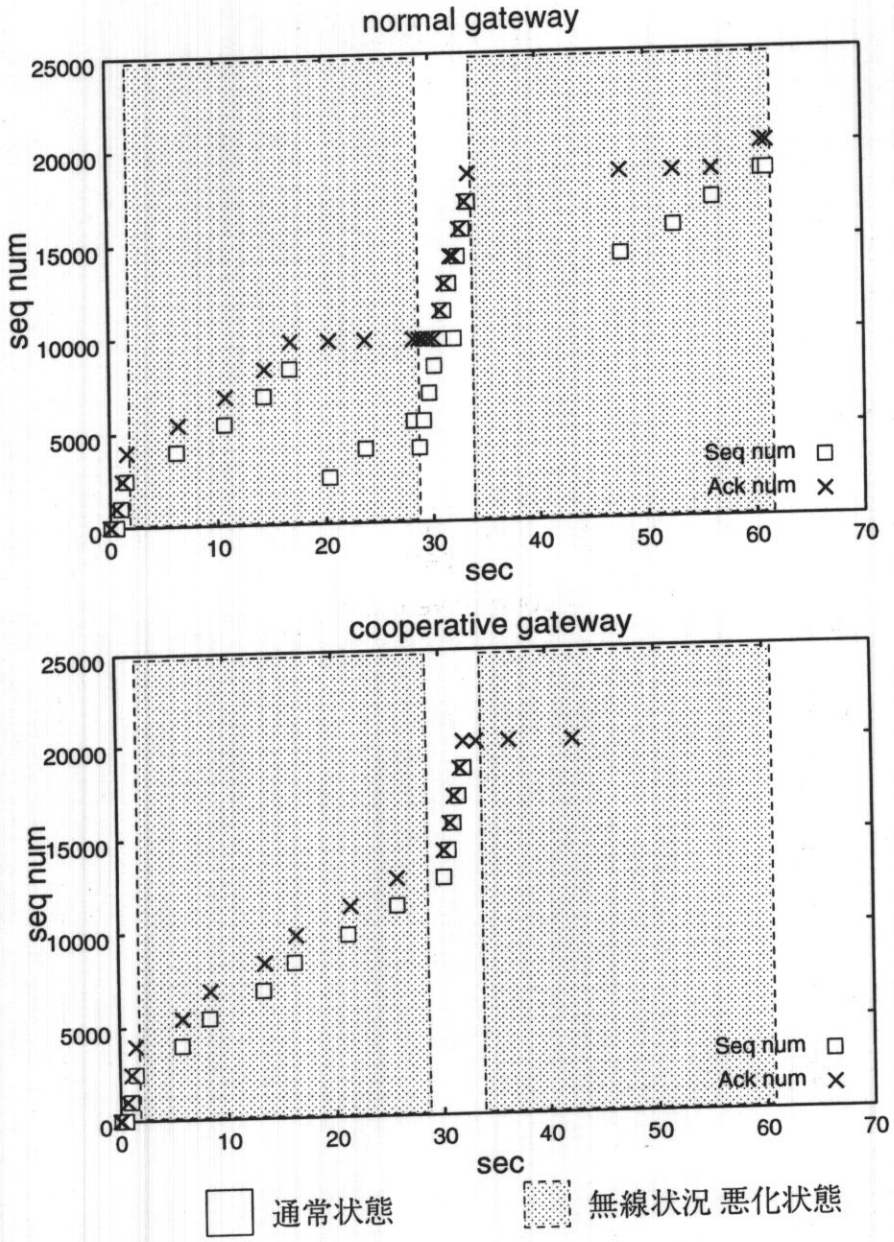


Figure 7.4 無線シミュレータの設定1でのTCPのタイムライン

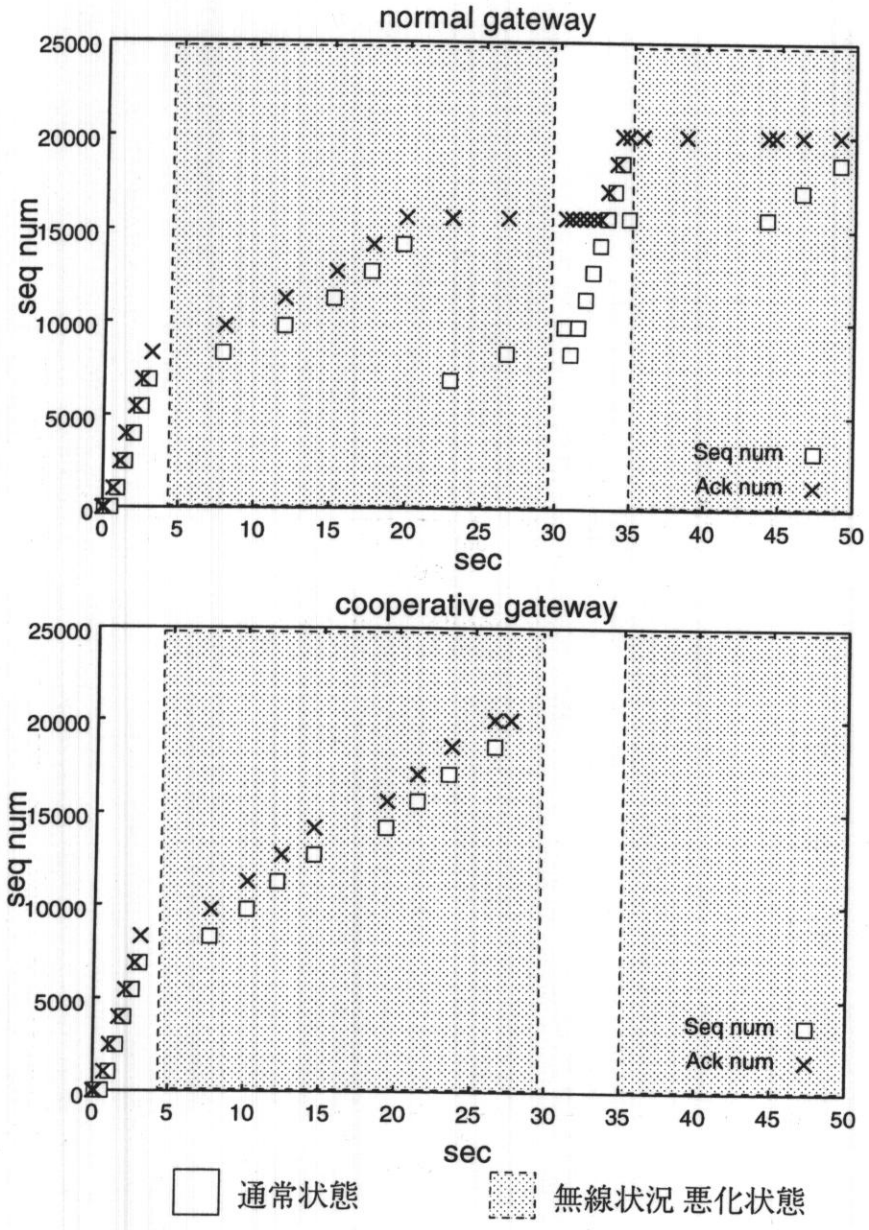


Figure 7.5 無線シミュレータの設定2でのTCPのタイムライン

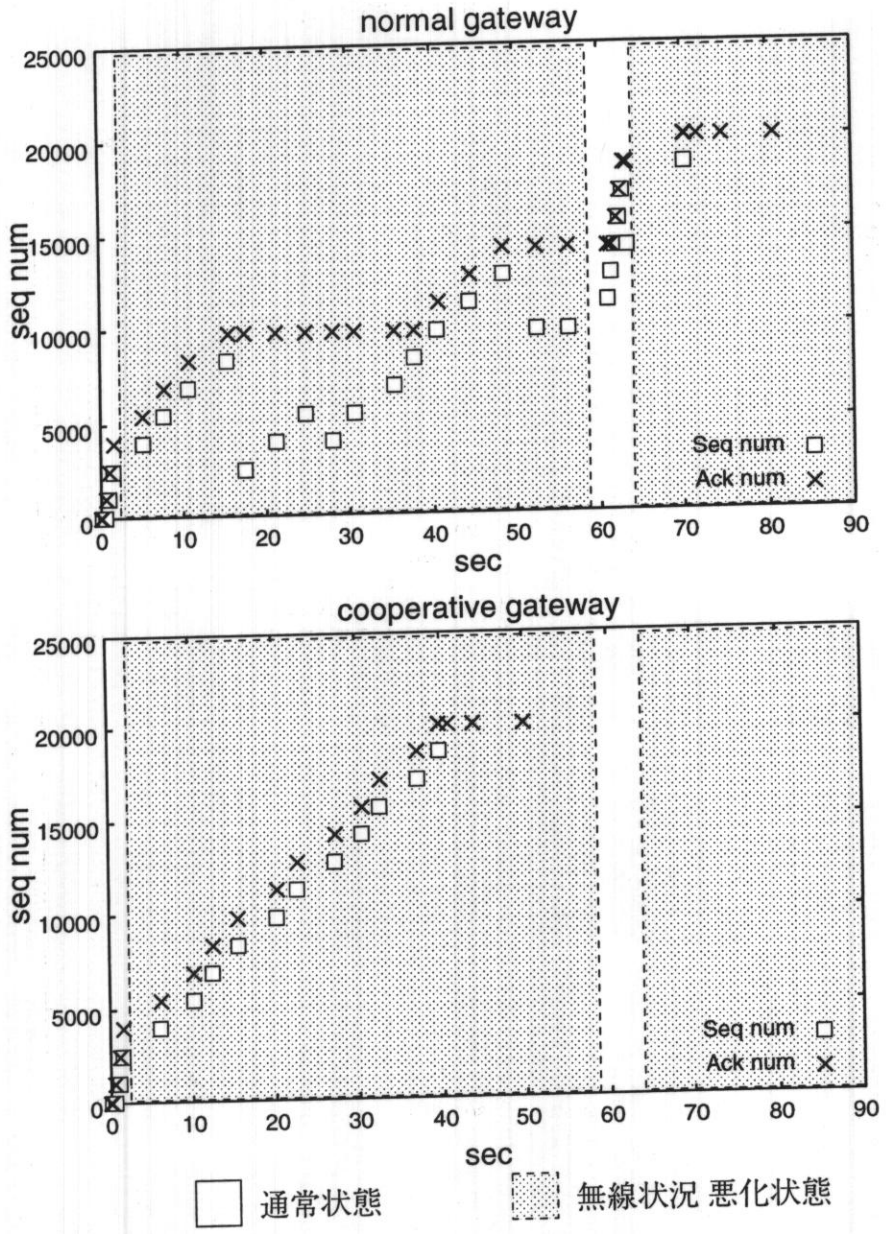


Figure 7.6 無線シミュレータの設定3でのTCPのタイムライン

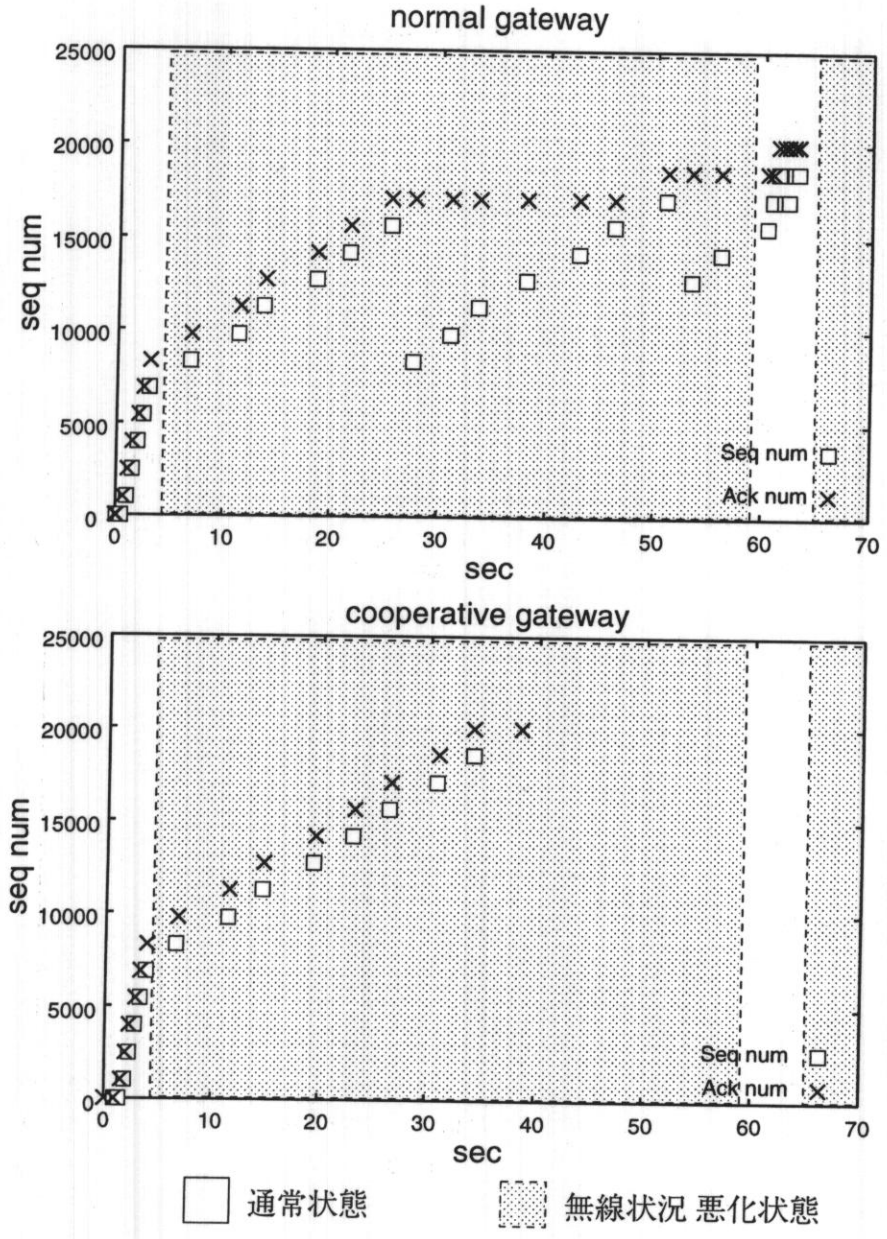


Figure 7.7 無線シミュレータの設定4でのTCPのタイムライン

## 7.3. 実際の携帯電話を用いた環境での測定

この実験では実際の携帯電話を使用した環境において、協調ゲートウェイの効果を測定した。無線状況を設定できる無線シミュレータを使用した場合と異なり、同じ無線状況での測定が困難であるため、実際の携帯電話を用いた測定では、通常のゲートウェイと協調ゲートウェイの動作の純粋な比較はできない。しかし、タイムラインを検討することにより、3章で述べたような不都合が解消されているかどうかの確認は可能である。

**測定環境**

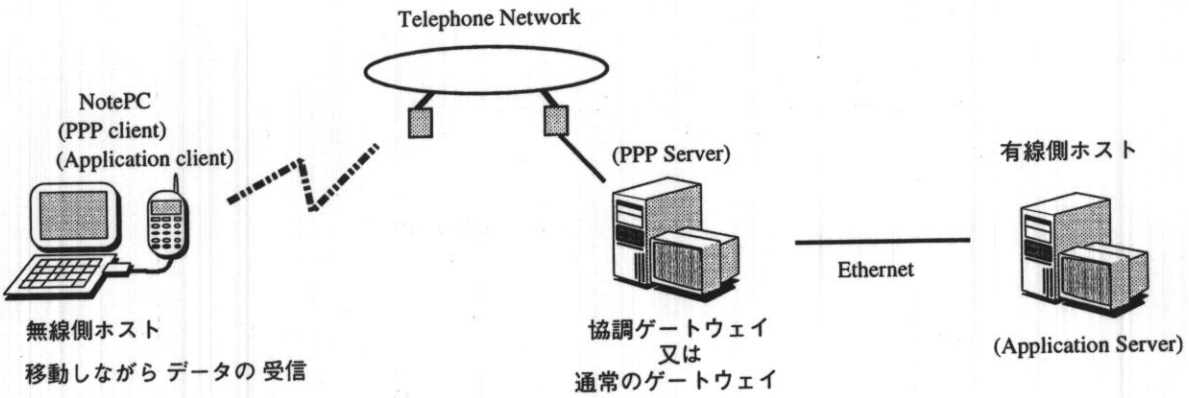


Figure 7.8 実際の携帯電話を用いた環境での測定

測定環境はfigure7.8のようになっている。無線シミュレータの測定実験のシミュレータ部分を実際の携帯電話を使用し、無線状況の悪い地域を移動しながら、有線側ホストから無線側ホストへの20kバイトのファイル転送を行った。

figure7.9は典型的な無線側ホストでのシーケンス番号とACK番号のタイムラインである。通常のゲートウェイを利用した場合には再送パケットが転送されていることがわかる。一方、協調ゲートウェイを利用した場合には再送パケットが転送されることなく通信が完了していることがわかる。

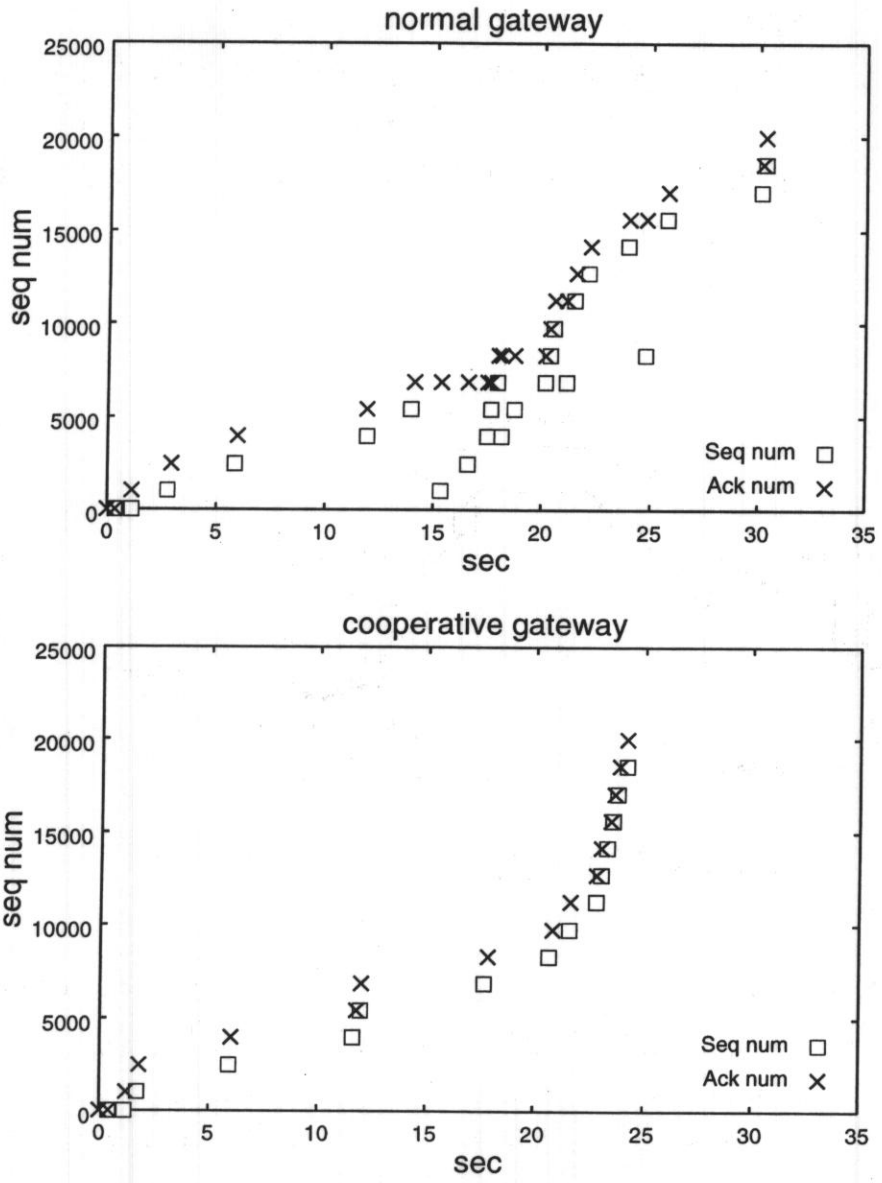


Figure 7.9 実際の携帯電話を用いた実験でのタイムライン

## 7.4. コネクション数を増加させた状況での測定

この実験では協調ゲートウェイを通過するTCPコネクションの数が増加した場合にどのような影響が生じるかを調べた。測定環境はfigure7.3と同じように無線シミュレータを用いた。但し、ゲートウェイ、無線側ホスト間の伝送速度は9.6kbpsと38.4kbpsの2種類で測定した。測定方法としては、有線側ホストから無線側ホストへの20Kバイトのファイル転送を行うTCPコネクションを同時に3本又は10本張り、全てのコネクションのデータ転送が終了するまでの時間を計測した。その際の協調ゲートウェイと通常のゲートウェイでのTCPの動作比較をした。

table7.1は無線区間の伝送速度9.6kbpsの時の測定結果である。

| | 3本 (9.6kbps) | 10本 (9.6kbps) |
|---|---|---|
| 通常ゲートウェイ | 28 sec | 68 sec (6本完了 4本タイムアウト) |
| 協調ゲートウェイ | 20 sec | 68 sec (6本完了 4本タイムアウト) |

Table 7.1 複数のTCPコネクションを確立した場合の協調ゲートウェイの有無による比較 (無線区間の伝送速度9.6kbps)

また、table7.2は無線区間の伝送速度38.4kbpsの時の測定結果である。

| | 3本 (38.4kbps) | 10本 (38.4kbps) |
|---|---|---|
| 通常ゲートウェイ | 7.3 sec | 63 sec (6本完了 4本タイムアウト) |
| 協調ゲートウェイ | 7.7 sec | 63 sec (6本完了 4本タイムアウト) |

Table 7.2 複数のTCPコネクションを確立した場合の協調ゲートウェイの有無による比較 (無線区間の伝送速度38.4kbps)

これらの結果からわかるように、同時にバルクデータ転送のコネクションを3本確立させた場合、無線区間の伝送速度9.6kbpsでは通常ゲートウェイを用いた場合よりも協調ゲートウェイを用いた場合の方が、早くデータ転送を完了させる

ことができた。逆に、無線区間の伝送速度38.6kbpsでは協調ゲートウェイを用いた場合、通常ゲートウェイを用いた場合よりもわずかではあるが、データ転送完了までの時間が長くなった。

また、同時にバルクデータ転送のコネクションを10本のように多数確立させようとすると、いくつかのコネクションは確立される前にタイムアウトを起こしてしまう。これは、ゲートウェイの問題ではなく、TCP自体の問題であると考えられる。TCPではスロースタートを採用しているため、ACKが返ってくることで、ネットワーク中に送り出すデータ量が増加する。そのため、最初に確立されるコネクションほどデータ転送量が大きく、無線環境のような小さな帯域では早い時間帯に確立されたコネクションによるデータ転送で帯域が埋められてしまい、後のコネクションはタイムアウトを起こしてしまうと考えられる。この問題を解決するためには、TCPそのものの見直しが必要であろう。

## 7.5. 考察

無線シミュレータを用いた実験、実際の携帯電話を利用した実験共に、通常ゲートウェイを利用した場合には3章で述べたように無線状況の悪化により、有線側ホストでタイムアウトを起こし、1度受信したパケットの再送が行われた。一方、協調ゲートウェイを利用した場合、5章で設計した通りに、無線状況が悪化した場合にも再送パケットが転送されることなく通信を継続することができた。

無線状況悪化時には無線区間のデータリンク層での誤り訂正による再送が頻発する。通常ゲートウェイを使用した際に生じるTCPのタイムアウトにより再送されたデータであっても、データリンク層ではその判断ができないために、受信側ホストによって破棄されるデータであっても、ビット誤りなどのエラーが起きている場合には誤り訂正を行い、確実に転送する。データリンク層での再送が頻発すると、上位層へデータが渡されるまでの時間が長くなる上に、既に受信したデータを再び受信することになるので、二重に効率が悪くなる。協調ゲートウェイを使用した場合にもTCPなどの上位層へデータが渡されるまでの時間は長くなるが、既に受信したデータを再び受信することはない。この点においてスループットの低下を抑えることが可能となった。このことより協調ゲートウェイは無

線状況の悪化が長引くような時に特に効果を発揮するといえる。

無線状況が回復するとデータは急激に流れるため、以前のACKの受信による再送や重複ACKによる再送も起きるが、回復も早いため通常ゲートウェイを使用している場合においてもタイミングによっては、協調ゲートウェイを利用した場合と時間的な差はそれほど生じることは無い。しかし、無駄なデータ転送は極力少なくすことが、有線環境に比べ小さな帯域しか持たない広域無線ネットワークにおいては重要である。その意味でも協調ゲートウェイは有効であるといえる。

# Chapter 8

# 今後の課題

本章では本研究において提案、設計、実装を通じて認識した今後の課題について述べる。

### スケーラビリティ

スケーラビリティの課題としては2種類が考えられる。1つはホスト数、コネクション数の増加に対するスケーラビリティ、もう1つは無線側ホストの先へのネットワークの拡大に対するスケーラビリティである。

1つ目のホスト数、コネクション数の増加に対するスケーラビリティに関しては、今回の実装ではむつかしい。それは、ユーザ空間で動作するiij-pppではカーネル内での処理に比べ遅いという問題が生じるからである。コネクションの増加に伴い、代理ACKの作成、コネクション管理ブロックの値の更新の負荷が増加する。そのためユーザ空間で処理を行っていては処理が遅れてしまう。コネクションの増加に耐えるためにはカーネル内モジュールとして協調ゲートウェイモジュールを組み込む必要があると考えられる。

また、今回の実装では有線側ホストからのデータ1つづつに対して代理ACKを返送していた。これではコネクションの増加、扱うパケットの増加に伴い、代理ACKの処理の増加が激しくなる。この問題に対しては1つ1つに対して代理ACKを返すのではなく、現行のTCPの遅延ACKのように数個分のパケットのACKを1つの代理ACKで済ますように改良し、代理ACKの処理の負荷を下げ

る必要がある。

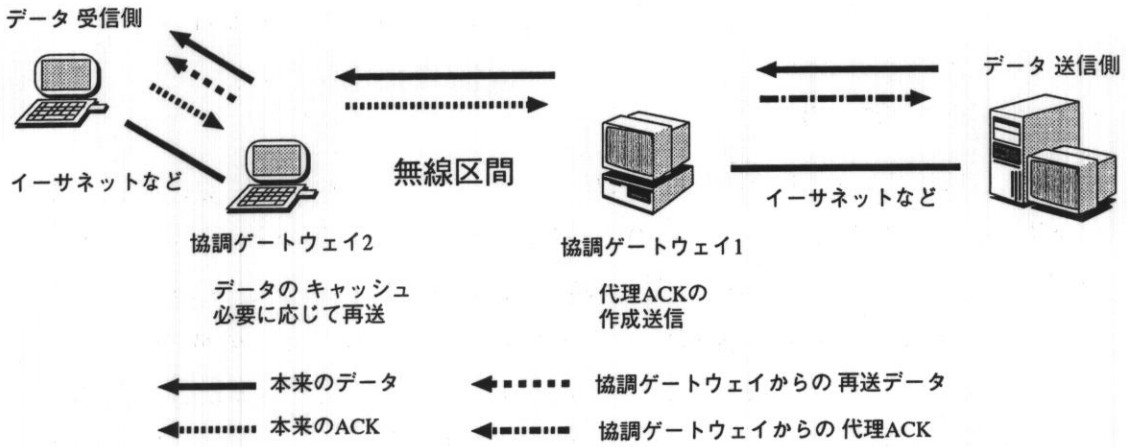


Figure 8.1 無線側ホストの先にデータ転送する場合の協調ゲートウェイの改良案

2つ目の無線側ホストの先へのネットワークの拡大に対するスケーラビリティに関しては、今回の設計・実装した協調ゲートウェイでは基本的に無線側ホストで終端するTCPコネクションを考慮して設計しているので改良が必要である。この場合、協調ゲートウェイと無線側ホストまでは広域無線通信メディアのデータリンク層の誤り訂正プロトコルによりエラーフリーが保証される。そのため、協調ゲートウェイで代理ACKを作成したデータに関しては、無線側ホストに転送するだけで、データそのものを保持しておく必要は無かった。しかし、無線側ホストでTCPコネクションが終端せずに、その先にネットワークが継っている場合には協調ゲートウェイに何らかの改良が必要になってくる。無線側ホストの先にイーサネットなどのデータリンク層での誤り訂正のないネットワークが継っている場合には、無線側ホストの先でデータ喪失が起きる可能性がある。このような場合、代理ACKが返っている有線側ホストからの再送がされないため、通信が滞る。

解決案としては、データリンク層で誤り訂正のあるネットワークの両端のゲートウェイを協調ゲートウェイにし、送信側ホストから見て奥の協調ゲートウェイ(figure8.1の協調ゲートウェイ2)でデータをキャッシュしておくことが考えられる。受信側ホストからのACKにより、協調ゲートウェイ2(figure8.1)では、キャッ

シュしているデータ破棄、または再送するように変更を加えれば良いと考えられる。このようなネットワーク環境での使用の需要と変更のコストとの兼ね合いになるであろうが、今後無線区間の帯域の拡大に伴い必要となる可能性はある。

### コネクションの切断、再接続

本方式では携帯電話、PHSがTCPコネクションが完了する前に切れてしまうような場合には対処していない。このような場合、2つの方針が考えられる。1つは協調ゲートウェイでタイムアウトを設定し、TCPコネクションをリセットしてしまう。もう1つは、再び無線側ホストと協調ゲートウェイとの間でPPPコネクションを確立し、通信の再開をする。前者は容易に解決できるが、根本的解決にはならない。後者は理想であるが解決しなければならない問題は多い。PPPの再接続の際に、同一ユーザの同一プロセスのコネクションであることを認証する機構や、再接続が困難な無線状況である場合にどれぐらいの時間まで協調ゲートウェイはデータを保持していなければならないのかといった問題がある。

またそれに付随して、Webなどのサーバがクライアントへ正常にデータが転送されたかを確認する必要のないアプリケーションの場合問題はないのだが、サーバが正常にデータがクライアントに送信できたことを確認する必要のあるアプリケーションの場合、問題が起きる可能性がある。協調ゲートウェイが代理ACKを送信しているので、送信したデータに対するACKが有線側ホストに返ってきている状況で、協調ゲートウェイと無線側ホストの間でデータ転送が完了する前に携帯電話が切れてしまったような場合には、有線側ホストは無線側ホストにデータが転送されたと判断してしまう可能性がある。

このような問題に対しては、今回の実装では協調ゲートウェイで代理ACKは作成するものの、無線側ホストの代わりにFINまでは作成しないので、FINによるコネクション終了しない場合には何らかの問題が発生したと判断するようにする必要がある。

### データロス

今回の設計・実装した協調ゲートウェイでは基本的に無線側ホストで終端するTCPコネクションを考慮して設計している。この場合は協調ゲートウェイと無線

側ホストまでは広域無線通信メディアのデータリンク層の誤り訂正プロトコルによりエラーフリーが保証されていたので、協調ゲートウェイでのデータのキャッシュは行っていなかった。データリンク層でのエラーフリーは保証されるが、上位層へデータを渡す際にバッファ不足等によるパケットロスが全く起きないというわけではない。今回は無線側ホストは終端でその先にネットワークが継っている想定であるため、広域無線通信メディアの転送速度を考慮するとバッファ不足によるパケットロスが起きる可能性は限りなく0に近いと考えられる。

しかし、より確実性の保証やネットワークの拡張性を保証するためにはスケーラビリティの項目で述べたように、協調ゲートウェイでのバッファリング、再送を実装することが望ましい。

### 協調ゲートウェイの応用

本研究は広域無線通信メディア利用時のTCPの性能改善に協調ゲートウェイを利用した。しかし、この協調ゲートウェイの考え方は、他の同様な特徴を持つデータリンク層を利用したネットワークを利用した場合にも応用できると考えられる。現行のTCPではパケットロスに関しては、全て輻輳と判断して輻輳回避を起動してしまう。これは輻輳以外の原因、ビット誤りや環境による影響でパケットロスが起きるようなネットワークでは輻輳回避で転送速度を下げることは得策ではない。そのようなネットワークでは協調ゲートウェイを用いることで、エンドホストのTCPによる輻輳回避の起動を避けることができ、極端なスループットの低下を防ぐことができると考えられる。

これらは今後の課題としたい。

# Chapter 9

# 結論

本研究では、携帯電話やPHSなどの広域無線環境においては、無線状況によりTCPのスループットが低下する原因についての検証、及びゲートウェイ協調による広域無線通信メディア利用時のTCPの性能改善を行った。

広域無線通信メディアではデータリンク層でのFEC、ARQによる誤り訂正により、無線状況が悪化し、データリンク層での再送が頻発するような状況では、TCPなどの上位層からみるとRTTが大きくなる。RTTが極端に大きくなることで、TCPでのタイムアウトにより不必要な再送が起きることが無線状況悪化時のスループットの低下に継っていた。

その検証を基に、有線側ホストゲートウェイで協調することで改善する方法を提案した。無線区間はデータリンク層の誤り訂正による信頼性の保証にまかせて、協調ゲートウェイから有線側ホストへ代理ACKを送信するようにした。この代理ACKを無線側ホストのウィンドウサイズを考慮したものにすることで、無線状況が悪化した場合、有線側ホストにデータの転送を止めさせることが可能になった。これによりTCPで起きる不必要なタイムアウト、データの再送を再送を防ぐことが可能となりスループットの低下を防ぐことができた。

# 謝辞

本研究の指導教官であり、素晴らしい研究室の雰囲気を整え、温かい御指導を頂いた奈良先端科学技術大学院大学の情報科学センターの湊 小太郎教授に心から感謝致します。

本研究の副指導教官であり、御指導を賜わりました奈良先端科学技術大学院大学の情報科学研究科の横矢 直和教授に心から感謝致します。

本研究を行うにあたり、大変参考になる御意見を頂き、親身になった御指導を頂いた奈良先端科学技術大学院大学の情報科学研究科の尾家 祐二教授に心から感謝致します。

本研究を行うにあたり、研究方針、論文指導、その他諸々の御指導を賜わりました奈良先端科学技術大学院大学の情報科学センターの砂原 秀樹教授に心から感謝致します。

また, 本研究を進めるにあたり、機材の提供や貴重なアドバイスを下さった慶応義塾大学環境情報学部の植原 啓介氏をはじめとする WIDE プロジェクトのみなさまに感謝致します。

TCP の内容を深く理解するための勉強会を開いて下さった奈良先端科学技術大学院大学情報ネットワーク講座の出水 法俊氏をはじめとする講座のみなさまに感謝致します。

入学当初わからないことだらけの私を基礎的なことから様々アドバイスを下さった研究室の先輩、現 NEC の細川 松寿氏には大変お世話になりました。心から感謝致します。

そして、論文執筆に当たり、御自身の忙しさを返り見ず、内容の吟味、適切なアドバイスを下さった情報科学センターの中村 豊氏には大変なご苦労をおかけしました。心から感謝致します。

研究活動を支えてるれた奈良先端科学技術大学院大学情報科学センター湊研究室のみなさまに感謝致します。

最後に、本研究を進めるにあたり、様々な配慮をして下さった家族に対して心より感謝致します。

# 参考文献


[1] R.Caceres, L.Iftode : "Improvimg the Performance of Reliable Transport Protocols in Mobile Computing Environments.",IEEE Journal on Selected Areas in Communications, June 1995

[2] J.Postel : "Transmission Control Protocol",RFC 793, September 1981

[3] J.Postel : "INTERNET PROTOCOL" RFC 791, September 1981

[4] V.Jacobson : "Congestion Avoidance and Control",Proc. ACM SIGCOMM '88, pp.314-329 CA, August 1988

[5] 杉山 泰一, 高槻 芳 : "生まれ変わる移動電話", 日経コミュニケーション 1998.6.15 No.272 pp.96-116, 1998/6

[6] 門脇 直人 : "衛星インターネット", bit vol.30 No.11 pp.87-94, 1998/11

[7] 田中 利憲, 藤井 敏孝, 服部 武 : "PHS マルチメディア伝送システム",NTT R&D,Vol.45,No.11 pp.1071-1078, 1996/11

[8] 松木 英生, 大野 友義, 高梨 斉, 田中 利憲 : "PHS における高品質・高速データ伝送に向けた誤り制御技術",NTT R&D,Vol.45,No.11 pp.1079-1088, 1996/11

[9] 桑原 守二 : "ディジタル移動通信", 科学新聞社 1992

[10] Kevin Fall, Sally Floyd : "Simulation-based Comparisons of Tahoe,Reno,and SACK TCP",Computer Communications Review, V. 26 N. 3, pp. 5-21. July 1996

[11] 三宅優, 長谷川輝之, 長谷川亨, 加藤聰彦 : "衛星インターネット用 TCP ゲートウェイの提案", 情報処理学会 マルチメディア通信と分散処理研究会 報告集 98-DPS-89 pp.63-68, 1998/6

[12] A.Bakre, B.R.Badrinath : "Handoff and system support for Indirect TCP/IP",In Second Usenix Symp. on Mobile and Location-Independent Computing, 1995/4

[13] A.Bakre, B.R.Badrinath : "I-TCP:Indirect TCP for Mobile Hosts", Proceedings of the 15th International Conference on Distributed Computing System(ICDCS), Vancouver,Canada, 1995/6

[14] Jani Kiiskinen, Markku Kojo, Mika Liljeberg, Kimmo Raatikainen, "Data Channel Service for Wireless Telephone Links",In the 2nd International Mobile Computing Conference, Hsinchu, Taiwan, ROC, March 1996.

[15] 植原 啓介, 西村 厚, 村井 純 : "LWPA:インターネット環境における広域無線通信メディア利用のためのアーキテクチャ", インターネットコンファレンス'97 論文集,pp49-62,1997/12

[16] M.Mathis, J.Mahdavi, "TCP Selective Acknowlegment Option",RFC2018, October 1996

[17] M.Mathis, J.Mahdavi : "Forward Acknowledgment: Refining TCP Congestion Control,", Proceedings of SIGCOMM'96, pp. 281-191, August, 1996, Stanford, CA.

[18] 細川 松寿, "無線環境に適応するネットワークプロトコルアーキテクチャの設計・実装に関する研究", 修士学位論文, 奈良先端科学技術大学院大学情報科学研究科 1998/2