

Title	通信メディアの特性を考慮したモバイルネットワークに関する研究
Author(s)	植田, 道成
Citation	
Issue Date	1999-03
Type	Thesis or Dissertation
Text version	author
URL	http://hdl.handle.net/10119/1252
Rights	
Description	Supervisor:中島 達夫, 情報科学研究科, 修士

修士論文

通信メディアの特性を考慮したモバイルネットワーク に関する研究

指導教官 中島 達夫 助教授

北陸先端科学技術大学院大学
情報科学研究科

植田 道成

1999年2月15日

目次

1	はじめに	1
2	多様な通信メディアと移動計算機環境	4
3	関連研究	8
3.1	IETF Mobile IP	8
3.1.1	IETF Mobile IP の概要	8
3.1.2	問題点	11
3.2	バークレースヌーププロトコル	11
3.2.1	バークレースヌーププロトコルの概要	11
3.2.2	問題点	14
4	JAIST Mobile IP システム	16
4.1	システムアーキテクチャ	16
4.2	IETF Mobile IP との相違	16
4.3	ソフトウェアの構成	19
4.4	RT-Mach への実装	21
5	通信メディア特性の変化への適応	24
5.1	最適化ポリシーの動的な変更	24
5.2	移動計算機 - プロキシエージェント間プロトコル	25
5.3	パケットスヌーパ	28
5.3.1	メディアセクタ内のスヌーパ	28
5.3.2	プロキシエージェント内のスヌーパ	29
5.3.3	最適化モジュールのインターフェース	30
5.4	最適化モジュール	31

5.4.1	パケット圧縮モジュール	31
5.4.2	エラーリカバリモジュール	34
6	評価と考察	38
6.1	最適化モジュールの評価	38
6.1.1	パケット圧縮モジュールの効果	38
6.1.2	エラーリカバリモジュールの効果	42
6.2	考察	46
7	おわりに	48

第 1 章

はじめに

本論文では、近年、多様化が進む通信メディアを有効に利用する柔軟な移動計算機環境について議論する。環境の基盤として JAIST Mobile IP システムを適用し、この枠組みの中で通信メディアの特性を考慮したモバイルネットワークを実現する機構の設計と実装について述べる。

移動計算機環境の発達には通信メディアの多様化に負うところが大きい。ユーザは携帯型の計算機を持ち運び、移動した先々で、有線のイーサネット、無線 LAN、公衆回線を利用した ISDN や PHS といった様々な通信メディアを利用してネットワークに接続することが可能である。しかし、ネットワークへの接続を物理的に確保できたとしても、再接続を行う際に、設定の変更や作業の中断を必要とするならば、ユーザにとって好ましくない状況である。携帯型計算機はサスペンド機能を備えているものが多く、電源を切っても再びレジュームすることによって、電源を切る前の状態に復元することができる。また、無線 LAN を利用していて通信範囲の外に出てしまった場合、もしくは、何らかの原因で電波が届かなくなってしまった場合、それでも作業を中断したくないときには、ユーザは携帯電話などを使用してネットワークへの接続を維持しようとするであろう。この際に、ネットワークアドレスの設定やアプリケーションの再起動、切断前の状態への復元といった作業を行わなければならないとすると、ユーザの負担はかなり大きくなる。

一方、単一の通信メディアを想定して、ネットワークアプリケーションのコネクションを維持した状態で計算機の移動を可能にするシステムは既に存在する。TCP/IP を利用したネットワーク上に移動計算機環境を構築する IETF Mobile IP [1] を実装したシステム [2] や、独自のプロトコルを用いて無線メディアの特徴を活かしたシームレスな計算機の移動を可能にするシステムなど、さまざまな方法が提案されている。しかし、通信メディアがこれほどまでに多様化した現在、単一の通信メディアに限定したシステムでは柔軟性

に乏しい。通信メディアの切替えを伴う計算機の移動を可能にし、なおかつ、アプリケーションのコネクションも維持でき、また、通信メディア特性の多様性にも対応したシステムが求められている。

こうした要求を十分に満足し、より柔軟な移動計算機環境を実現するために提案された枠組みが JAIST Mobile IP システムである。JAIST Mobile IP システムは以下の4つの問題に対して明確な解決法を提示する。

- コネクションを維持した状態での計算機の円滑な移動
- 適切な通信メディアへの切り替え
- 通信メディア切り替えのタイミングの制御
- 通信メディア特性の変化への適応

従来のシステムはこれらの問題に個別に対応しようとしていたが、JAIST Mobile IP システムは、これらを統合的に扱う枠組みを提供する。また、JAIST Mobile IP システムは、IETF Mobile IP を基盤としており、既存の TCP/IP を用いたネットワーク上でそのまま利用できることも大きな利点である。本論文では、とくに第4の問題「通信メディア特性の変化への適応」に焦点を当て、メディア切替や計算機の移動によって発生する様々なメディア特性の変化、すなわちエラー率、バンド幅、遅延などの変化に適応する機構の設計と実装について議論する。

各通信メディアの特性に合わせた最適化の研究は、従来から数多く行われてきた。しかし、それらの方法を固定的に用いた場合、特性の異なるメディアに変更したときの性能は保障されず、場合によってはその最適化が性能低下の要因となることもあり得る。JAIST Mobile IP システムにおいては、まったく特性の異なる通信メディアを動的に切替えて使用することを前提としているため、従来の方法を固定的に組み込むことは好ましくない。

そこで、本論文では、こうした既存の最適化手法を状況に応じて使い分ける機構を提案する。メディアの特性に特化した最適化手法を動的に変更可能なモジュールとして作成し、これらを特性の変化に応じて切り替えることで、移動計算機環境におけるパケット転送の効率を最大限に向上する。最適化モジュールには統一されたインタフェースを提供し、新しいモジュールの追加、変更は容易である。また、従来の方法ではしばしば最適化のためにネットワークの構成をプロトコルも含めて全体的に変更する必要があった。本論文で提案する方法は、移動計算機宛てのパケットのスヌーピングをベースに最適化の処理を行うことで、既存のネットワークに影響を与えることなく性能向上を実現する。

最後に、以上の基本設計をもとに実際に稼動するシステムを構築し、このシステムに対して評価を行うことで、本論文において提示した方法が有効であることを示す。

以下、第2章では、現在の移動計算機環境に求められているものと、それを実現するために何が障害となるのかについて議論し、本論文で提案する方法の必要性を示す。第3章では、2章で提示した問題点を克服するためにこれまでに行われてきた研究について示し、それらの研究が解決した問題といまだ不十分である点について議論する。第4章では、JAIST Mobile IP システムの枠組みを設計と実装の観点から述べ、本論文が提案する通信メディア特性の変化への適応が JAIST Mobile IP システムのどの部分に相当するのかわを示す。第5章では、通信メディアの特性変化への適応を可能にする機構が実際にどのように機能するのかについて、その設計と実装を詳細に述べ、サンプルとして作成した2種類の最適化モジュールを紹介する。第6章では、JAIST Mobile IP システムを用いた実測の評価を行い、最適化の効果を示す。同時に残された課題の検討も行う。第7章では、まとめと今後の課題について述べる。

第 2 章

多様な通信メディアと移動計算機環境

計算機ハードウェアの小型化と通信メディアの多様化によって、時間や場所を選ばずネットワーク上のさまざまなサービスや資源にアクセスできる移動計算機環境が実現している。通信メディアとしては、有線 LAN、無線 LAN、公衆回線を利用した ISDN や PHS といったさまざまなものが利用可能である。ユーザはノート型計算機などを携帯し、移動した先々で有効な通信メディアを選択してネットワークへの接続を維持することができる。

一方、ネットワークに接続された計算機で利用されるアプリケーションは、クライアント・サーバ型の形態を取るものが一般的である。これらのアプリケーションは、クライアントとサーバの間でコネクションを必要とするものが多く、またクライアントがサブネットを越えて移動することは想定していない。携帯型計算機の普及に伴って、大学の構内やオフィスビル内で携帯端末からこうしたクライアント・サーバ型のアプリケーションを利用するケースが増えているが、計算機を持ち運ぶことができ、移動先でネットワークに再び接続できたとしても、ユーザは作業を中断せざるを得ないのが現状である。

コネクションの切断や設定の変更を必要としないクライアントの移動を、アプリケーション側でサポートする試みもなされているが、個々のアプリケーションで計算機の移動に対応して行こうというアプローチは非効率的である。

こうした状況を受けて、近年、システムレベルで計算機の移動を透過にするアプローチへの期待が高まっている (図 2.1)。このようなシステムの実現によって、ユーザは移動の際の作業の中断や復元を行う必要がなくなり、また煩雑なネットワークの設定からも解放される。さらに、アプリケーションレベルで計算機の移動を考慮する必要が無くなるため開発・運用面での有用性も期待できる。また、クライアントの位置的な制約が緩むことで、グループウェアなどのアプリケーションの分野において今までにはない形態のものが

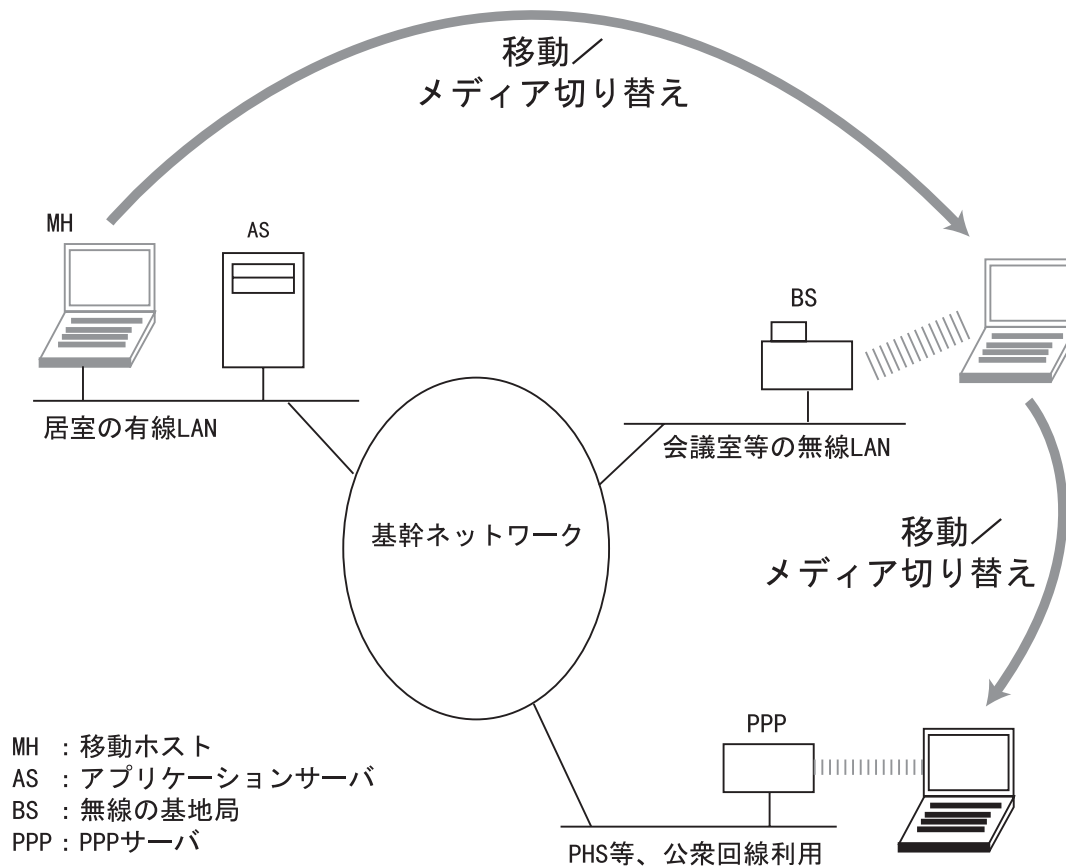


図 2.1: 移動計算機環境

登場する可能性もある。

しかし、このような環境を実現するためには、解決しなければならない問題が多く残されている。とくに、以下で示す4つの問題点は、柔軟な移動計算機環境を構築する上で根本的に解決しなければならないものである。

計算機を識別するためのアドレス

現在の TCP/IP に代表されるネットワークプロトコルは、通信メディアやネットワークの接続位置が恒常的に変化しない計算機環境を想定して設計されたものである。したがって、計算機が移動し、使用する通信メディアやネットワークとの接続位置がその時々によって変化することなどは考慮していない。その顕著な例は、パケットを配送する際の宛先を示す IP アドレスが計算機の識別子としては機能しないという特徴に現れている。つまり、TCP/IP において IP アドレスは計算機とネットワークの接続点の識別子

として機能している。計算機のネットワークインターフェースとサブネットの端子の一つを結び付けた時に初めて、有効な IP アドレスが割り当てられるのである。したがって、サブネットをこえた計算機の移動は、この組み合わせの変更を強いるため、TCP/IP だけでは計算機の移動をサポートすることができない。

多様な通信メディア

図 2.1 のような環境を想定した場合、移動計算機は複数の通信メディアを常時利用することが可能である。この環境を活用するためには計算機の動作中にそれらを円滑に切り替える仕組みが必要不可欠である。

メディア切り替えのタイミング

無線ネットワークにおけるハンドオフ処理に関しては様々な最適化が提案されているが、異なる通信メディアの切り替えを考慮したものはない。また、円滑な通信メディアの切り替えを可能にするには、使用できるメディアの情報を統一して管理し、その情報をもとに切り替えのタイミングを適切に制御する必要がある。

メディア特性の変化

通信メディアの動的な切り替えとそのタイミングの最適化を可能にするだけでは、十分な通信性能を得ることはできない。これは、通信メディアの切り替えや移動先のロケーションによって、バンド幅、エラー率、遅延などの特性が大きく異なるからである。また、既存の最適化の方法は、それぞれの通信メディアに特化したものが多く、これらを固定的に用いていたのでは、通信メディアの動的な切替を想定する場合、切替後の通信メディアに対応しきれないどころか、かえって性能を低下させてしまう。

たとえば、無線メディアに特化した最適化の方法として、移動計算機と無線の基地局の間でローカルなエラーリカバリを可能にするシステムが考えられる。しかし、計算機をエラーの少ない有線の通信メディアでネットワークに接続するときには、その最適化は単にオーバーヘッドとなるだけである。また、PHS 等の通信メディアを用いる場合、狭いバンド幅を有効に利用するためにパケットの圧縮を行う最適化が考えられる。しかしこの場合も、十分なバンド幅を持つメディアに切替えたときには、パケットを圧縮するために必要となる時間がバンド幅の有効利用をかえって妨げてしまう。したがって、通信メディアを自在に切り替えることができる環境を想定した場合、各メディアの特性に応じて通信の性能を向上するための機構を用意し、それらを状況に応じて動的に切り替える仕組みが必要と

なる。

以上の4つの問題に焦点を当て、これらを解決するために提案された枠組みが JAIST Mobile IP システムである。とくに、本論文において議論の対象とする「通信メディア特性の変化への適応」は、通信メディア間の特性の違いが大きくなっている現状で、それらを切替えて利用することを想定した場合、非常に重要であると考えられる。

第 3 章

関連研究

本章では、JAIST Mobile IP システム以前の計算機の移動をシステム側でサポートする研究、および通信メディアの特性をカバーしパケット転送の最適化を行う研究について代表的なものを紹介する。

3.1 IETF Mobile IP

計算機の移動透過性を実現する枠組みとしてとして IETF Mobile IP がある。しかし、この枠組みに則して実装された従来のシステムでは、本稿で想定する図 2.1 のような環境を実現することはできない。本節では IETF Mobile IP の概要と、従来の Mobile IP システムでは解決し得ない問題について述べる。

3.1.1 IETF Mobile IP の概要

IETF Mobile IP の基本アーキテクチャを図 3.1 に示す。

図において、R0、R1、R2、R3 はルータであり、各サブネットは、それぞれルータを経由してバックボーンに接続されている。また、この図において M は移動ホスト、S は M と通信する固定計算機を示している。IETF Mobile IP では、各移動計算機は、その計算機のホームネットワーク上に自分の現在の IP アドレスを管理してもらうホームエージェントを持っている。ホームエージェントは、移動計算機がホームネットワークから離れている間に届いた IP パケットを移動ホストに転送する。図 3.1 では、ルータ R1 が移動計算機 M のホームエージェントであり、そのルータがあるサブネットが M のホームネットワークである。各移動計算機には、ホームアドレスと呼ばれる一定不変の IP アドレスが割り当てられる。このホームアドレスは、ホームネットワーク上の IP アドレスで、他の

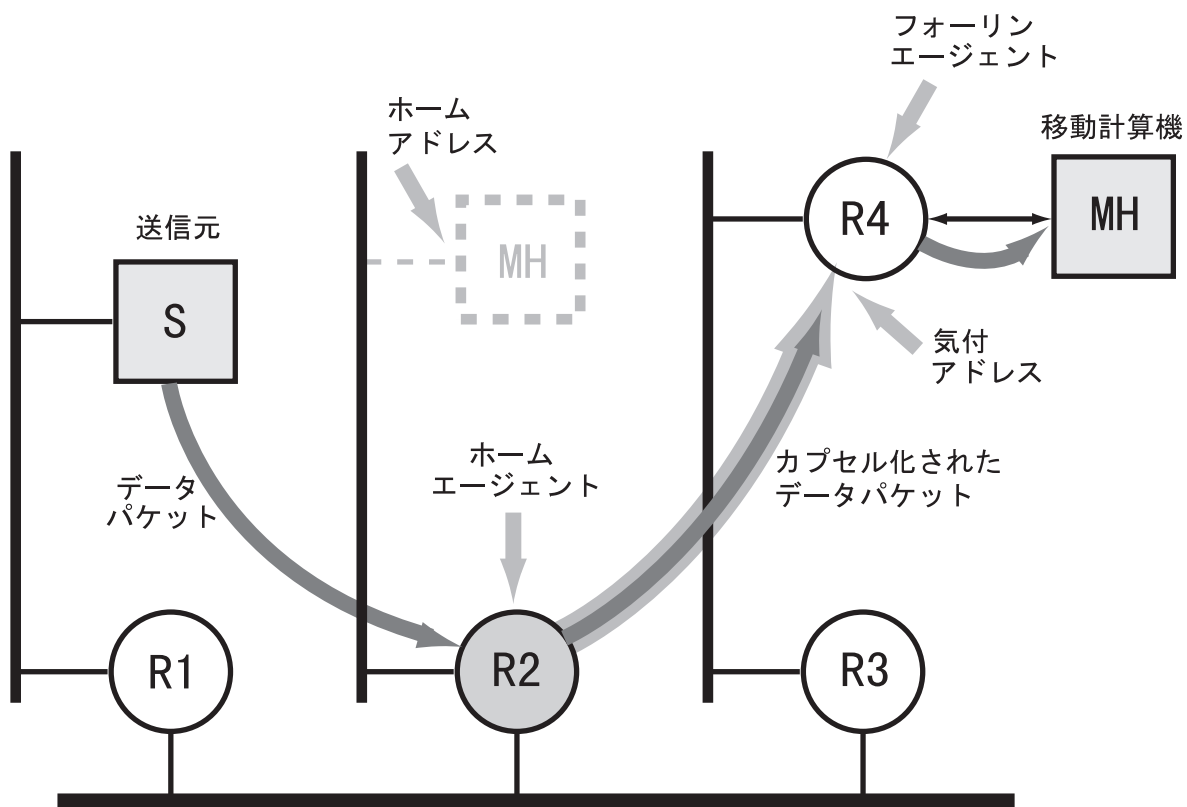


図 3.1: IETF Mobile IP の基本アーキテクチャ

計算機から移動計算機を識別するために使用される。ホームアドレスの経路情報のアナウンスは、ホームエージェントによって行われる。

ホームネットワークから離れた移動計算機は、フォーリンエージェントが存在するサブネットに接続することができる。そして、このサブネットに接続するとき、気付アドレスと呼ばれる IP アドレスが割り当てられる。気付アドレスは、移動ホストが接続したサブネットによって異なり、その割り当てには2つの方法がある。1つは、接続したサブネット上にあるフォーリンエージェントの IP アドレスを使用する方法で、もう1つは、DHCP[3] などを用いて割り当てられた一時的に使用可能な IP アドレスを気付アドレスとして使用する方法である。移動計算機は、このフォーリンエージェントによって割り当てられた IP アドレスをそのホストのホームエージェントに登録する。そして、ホームエージェントがその登録された IP アドレスに IP パケットを転送することで、移動計算機に IP パケットが配送される。図 3.1において、ルータ R3 がフォーリンエージェントであり、ホームエージェント R1 から転送されてきた IP パケットを移動計算機 M に配送する。IETF Mobile IP では、既存のインターネットへの変更を最小限に抑えるために、ホームエージェントとフォーリンエージェント間の通信は、IPinIP のトンネリング [5] を用いて行われる。すなわち、図 3.1において、固定計算機 S から移動計算機 M に対してデータ転送を行う場合には、以下のような手順となる。

1. ホスト S は、M のホームアドレスを指定して IP パケットを送る。
2. ホームエージェント R1 は、M のホームアドレスの IP パケットを受け取ると、その IP パケットをホームエージェントに登録されている気付アドレスをもとに M が接続されたサブネット上のフォーリンエージェント R3 に IPinIP でカプセル化し転送する。
3. R3 は、カプセル化された IP パケットからもとの IP パケットを取り出し、ホスト M にその IP パケットを送る。

計算機 M がホームネットワークから移動して他のサブネットワークに接続するとき、M は、フォーリンエージェントを見付ける必要がある。これを実現するためには、Agent Discovery プロトコルが使用される。このプロトコルには、ICMP ルータ広告メッセージを拡張したものが用いられている。そして、計算機 M は、サブネット上のフォーリンエージェントを発見し、気付アドレスが割り当てられると、その気付アドレスをその計算機のホームエージェントに登録する。そして、ホームエージェントは、登録された気付アドレスへの転送準備が完了すると、ホームエージェントに送られた IP パケットを計算機 M に転送することが可能となる。

3.1.2 問題点

従来の IETF Mobile IP の実装には、図 2.1 のような環境を実現する上で解決しなければならない問題点が残されている。

本来、IETF Mobile IP は、計算機の移動を透過にするための標準的なネットワークアーキテクチャの確立を目指して提案されたものである。ただし、IETF Mobile IP の枠組みには、動的な通信メディアの切替やそれに伴う複雑なハンドオフ処理、また通信メディアの特性に合わせた最適化といった項目は明確に含まれていない。IETF Mobile IP を実装した従来のシステムは、単一の通信メディアの使用を想定した環境では、十分に機能するが、複数のメディアを使い分ける環境を想定する場合、相当の拡張が必要となる。また、そうした環境に合わせて、枠組み自体の設計を見直す必要もある。

3.2 バークレースヌーププロトコル

一方、無線メディアの特性を利用して、計算機の移動透過性を実現し通信の最適化までを考慮した研究として、バークレースヌーププロトコル [7] がある。このシステムでは、無線セルの特性を利用した円滑な計算機の移動を可能にし、無線リンクにおける TCP の性能改善を IP レベルのパケットスヌーピングによって実現している。以下では、バークレースヌーププロトコルの概要とその問題点を示す。

3.2.1 バークレースヌーププロトコルの概要

バークレースヌーププロトコルは無線ネットワークを対象にしたプロトコルである。実際に、移動計算機環境においては、無線リンクが不可欠な要素となっているが、その無線リンクの特徴としては、基地局の電波が届くところであれば、場所を選ばずにネットワークに接続できる、リンクを確立する基地局を切替えるだけで、異なるネットワークに接続し直すことができ、計算機の移動を簡単にするという利点がある。

一方、イーサネットなどの有線リンクと比較した場合、バンド幅が狭く、エラー率が高い、また、無線の電波到達範囲を超えることによる一時的な切断が発生するといった弱点が挙げられる。ネットワークプロトコルとアプリケーションはこれらを扱わなければならない。加えてユーザのモビリティとハンドオフを扱わなければならない。ここで、ハンドオフとは、コミュニケーションステートを基地局から別の基地局へ転送することを示す。ハンドオフ処理はしばしば、パケットの損失や様々な遅延を引き起こし、これをいかに小さく抑えるかが、無線ネットワークを用いた移動計算機環境を構築する上で重要な課題とな

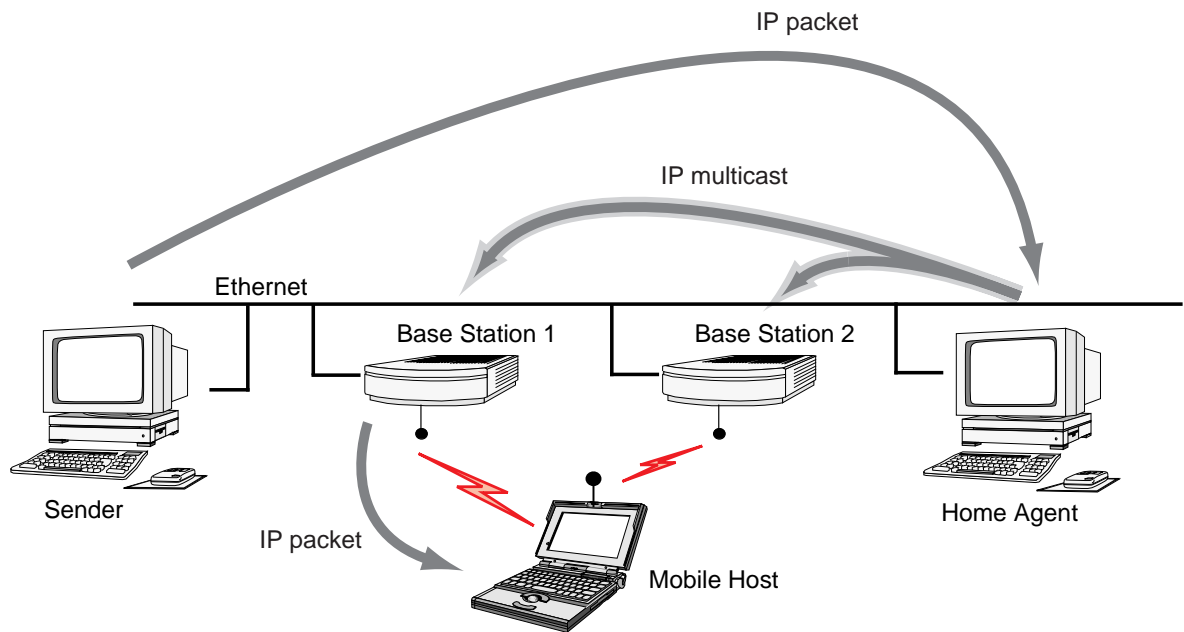


図 3.2: バークレースヌーププロトコルのネットワーク構成

る。バークレースヌーププロトコルは、既存のネットワーク上の固定計算機の実装に手を加えずに、またアプリケーションを再コンパイルすることなしに、ワイヤレスリンクネットワークのパフォーマンスを改善することを目標としている。

バークレースヌーププロトコルが想定するネットワークの構成を図 3.2 に示す。バークレースヌーププロトコルにおいても、IETF Mobile IP と同様に、移動計算機へのパケットの転送はホームエージェントを介して行う。しかし、そのルーティングプロトコルは IETF Mobile IP のそれとはかなり異なり、無線ネットワークの特性を最大限に活用したものである。

ユーザの移動によるパケットの損失やリンクの切断、遅延の増大は望ましくないが、無線システムのセル間を計算機が移動する場合、どうして固定計算機と移動計算機の間ルーティングの情報を更新しなければならない。このために、ハンドオフ処理が必要となるわけだが、IETF Mobile IP のように、通信メディアに依存しないプロトコルにおいては、計算機の移動中のパケット損失や長い遅延は、ある程度仕方のないものと考えられる。有線メディアでネットワークに接続しているときには、次にどこに移動するかといったことを予測することは不可能である。通信メディアを無線リンクに特定した場合、この予測がある程度可能となる。そこで、この特性を活かして、バークレースヌーププロトコルでは、移動計算機の近隣の基地局に対するマルチキャストとインテリジェントなバッ

ファリングを用いてパケットの損失をなくし、パフォーマンスの低下を防ぐ。

バークレースヌーププロトコルのスキームでは、移動計算機から固定計算機に向かうパケットはIETF Mobile IP 同様、通常のIP ルーティングサポートを利用する。移動計算機に向かうパケットのルーティングは基本的に以下の3つの部分から構成される。

1. Mobile IP のホームエージェントと同様に、計算機の移動を管理すること。
2. 移動計算機の物理的なロケーションを決定すること。
3. ホームエージェントから移動計算機へのパケットの配送をサポートすること。

意味的にはIETF Mobile IP のそれとほぼ同じであるが、実現の方法がかなり異なる。第1段階は、ホームエージェントによるパケットの転送である。IETF Mobile IP と同様に、移動計算機のホームアドレスに宛てられたパケットはすべて、ホームエージェントによって取り込まれる。各移動計算機には移動時にテンポラリなIP マルチキャストアドレスがアサインされており、ホームエージェントは移動計算機宛てのパケットをカプセル化し対応するマルチキャストグループに転送する。このマルチキャストのメンバーは移動計算機の周辺に位置する基地局を含んでいる。図3.2においては、基地局1と2がこれに相当する。

第2段階は、移動計算機の現在位置の決定である。各基地局は電波到達範囲内の全ての移動計算機に対して定期的にビーコンメッセージをブロードキャストしている。移動計算機はこのビーコンの記録を取っており、これを利用して最も通信状態の良い基地局を選択する。また、移動計算機はどの基地局のセルに入るべきかを判断し、ハンドオフの予測を行う。この決定に基づいて、移動計算機はホームエージェントと様々な基地局の間のルーティングを設定する。

第3段階は、第2段階のルーティングに基づいてホームエージェントから移動計算機にパケットを配送する。セル内に移動計算機を持つ基地局は、その移動計算機に対応するIP マルチキャストのグループに参加している。マルチキャストグループ上のホームエージェントから転送される各パケットは、基本的にこの基地局によって移動計算機にフォワードされる。3.2の状況では、基地局1が選ばれている。また、ハンドオフの対象となっている基地局は、ハンドオフされようとしている移動計算機を受け入れられるように、対応するマルチキャストグループへの参加を求められる。これらの基地局は、マルチキャストグループのパケットを無線リンクには流さないが、そのかわりにホームエージェントからの最後のいくつかのパケットをバッファリングする(3.2の基地局2の状態)。移動計算機がハンドオフされると、つまり、移動計算機がセルの中に入ってくると、その基地局が

バッファリングしていたパケットを転送しはじめる。このハンドオフスキームは、ハンドオフ時のデータロスを最小限に抑える。あらかじめルーティングをセッティングすることでこれを可能にしていると言える。これによってTCPのパフォーマンスに影響を与えることはない。

さらに、バークレースヌーププロトコルは、無線リンクのエラーをリカバリーし、TCPのパフォーマンスを改善するメカニズムも提供している。従来のTCPはパケット損失の主原因が輻輳にあるような有線ネットワークで十分に機能するように最適化されている。一方、無線リンクを利用したネットワークのパケット損失は、多くの場合、ビットエラーによって生じる。このような特徴を持つネットワークで従来のTCPをそのまま用いると、たとえば輻輳回避や高速リカバリーのアルゴリズムが不必要に働いてスループットを低減させてしまう。そこで、バークレースヌープでは、無線の基地局にエラーリカバリを行う機能を持たせ、移動計算機と無線基地局間でローカルな再転送を可能にしている。

基本的にはTCPのデータパケットをキャッシングし、確認応答パケットを監視することで失われたパケットのローカルな再転送を可能にする。移動計算機へ向かうデータのエラーリカバリには、基地局内のエラーリカバリモジュールに、移動計算機から確認応答が届いていないパケットのキャッシングを行う機能と、移動計算機からの確認応答を監視する機能を持たせることで対処している。TCPは順番が狂って到着したパケットに対して重複確認応答を返す。そこで、基地局のエラーリカバリモジュールは、移動計算機から返された重複確認応答をもとに失われたパケットを特定し、データパケットキャッシュの中から対応するパケットを取り出して再転送を行う。また、移動計算機から送信されるデータのエラーリカバリには、基地局のエラーリカバリモジュールにTCPデータのシーケンス番号を監視させることで対処する。パケットのロスを検出した場合、移動計算機のTCPに対して選択的確認応答を送り、対応するパケットを再転送させる。

以上のメカニズムによって双方向のパフォーマンス向上を実現している。

3.2.2 問題点

バークレースヌーププロトコルは、完全に無線メディアに特化したシステムである。したがって、バークレースヌープで提案されているハンドオフスキームは、移動先やロケーションによって使用する通信メディアが変わってしまう図2.1のような環境では使用できない。移動計算機が有線の通信メディアを用いてネットワークに接続している場合、その計算機が次にどのサブネットに移動しそうかなどといった予測を行うことは不可能である。

また、パフォーマンスを最優先させた実装にも問題がある。パケットコピーのオーバーヘッドを避けるために、カーネルのパケットバッファに変更を加えており、最適化ポリシーの変更や追加は非常に困難である。たしかに、パケットロスが発生しないときには、通常の TCP/IP と同等のパフォーマンスを実現しているが、柔軟性および拡張性を犠牲にしている。

第 4 章

JAIST Mobile IP システム

本章では、従来のシステムの問題点を解決し、より柔軟な移動計算機環境を実現する JAIST Mobile IP システムの枠組みを示す。

4.1 システムアーキテクチャ

JAIST Mobile IP の基本アーキテクチャを図 4.1 に示す。

基本的な構成は IETF Mobile IP と同様であるが、フォーリンエージェントに代わるものとしてプロキシエージェントを配置している点が主に異なる。この図において、R1、R2、R3、R4 はルータであり、それぞれのサブネットはこれらのルータを経由して基幹ネットワークに接続されている。R2 はホームエージェント、PA はプロキシエージェントであり、移動計算機 (MH) はこの二つのエージェントのサポートを受けることで、サブネットを越えた移動や通信メディアの切り替え、メディア特性の変化への適応といった機能を実現する。ホームエージェントの機能は、IETF Mobile IP に準ずる。また、移動計算機宛てのパケットの流れも IETF Mobile IP と同様である。例外としては、移動計算機とプロキシエージェント間をカプセル化されたパケットが流れる点である。移動計算機がホームネットワーク以外のネットワークに接続する場合、そのネットワーク上のプロキシエージェントを利用することで、通信メディアの特性の変化に適応することができる。

4.2 IETF Mobile IP との相違

次に、JAIST Mobile IP の特徴を IETF Mobile IP との比較によって示す。

f

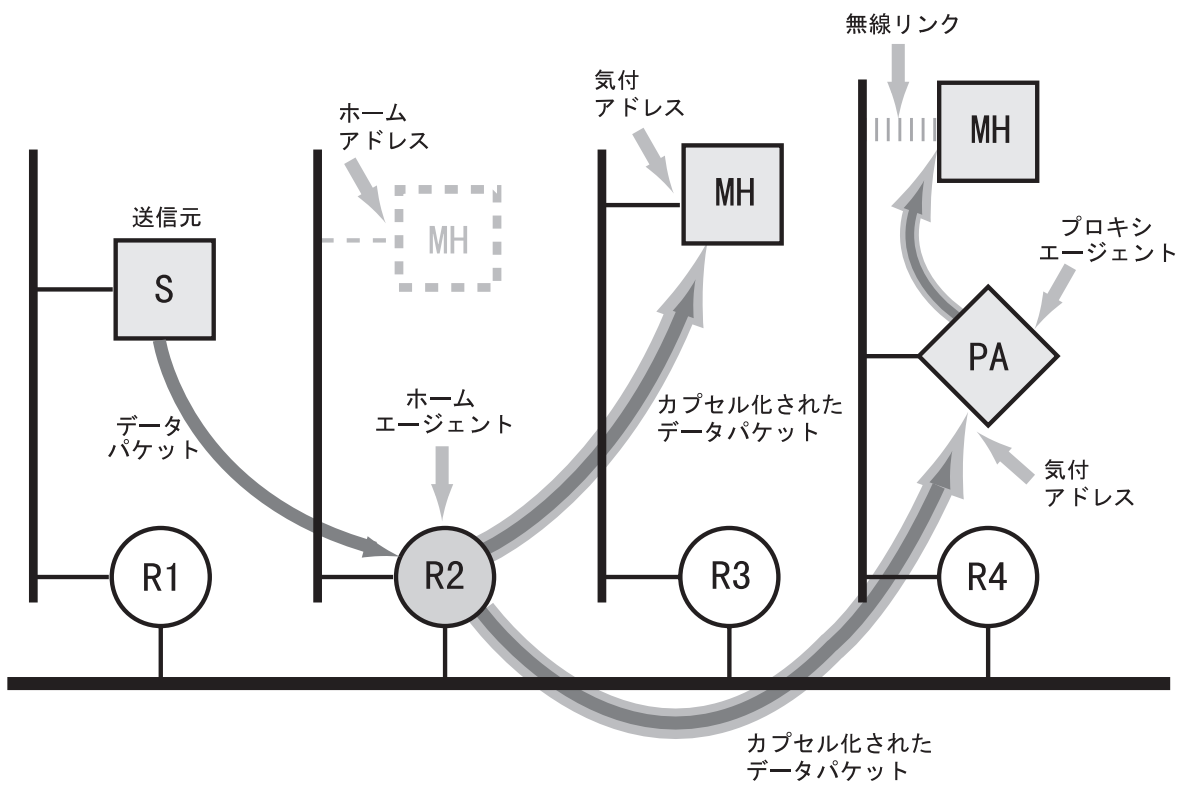


図 4.1: JAIST Mobile IP システムの基本アーキテクチャ

ホームエージェント

計算機の移動については IETF Mobile IP と同様に扱う。また、移動計算機における通信メディアの切替はホームエージェントから見ると計算機の移動と同様に扱うことができる。したがって、JAIST Mobile IP では IETF Mobile IP のホームエージェントをそのまま利用する。

プロキシエージェント

IETF Mobile IP ではフォーリンエージェントを用いることで、移動した計算機へのパケット転送を可能にしている。JAIST Mobile IP では、このフォーリンエージェントを拡張したプロキシエージェントを提供する。プロキシエージェントは移動計算機とネットワークとの末端の回線を結ぶ計算機、すなわち PPP サーバや無線 LAN の基地局を管理する計算機である。移動計算機は移動先のネットワーク上のプロキシエージェントと協調し、使用している通信メディアの特性に応じてパケットの転送を最適化する。具体的には、移動計算機とプロキシエージェントの間で最適化ポリシーを動的に変更するためのプロトコルを設け、様々な最適化ポリシーを通信メディアの特性に応じて使い分ける。したがって、最適化の対象となる部分は移動計算機とネットワークをつなぐ末端の回線に絞られる。しかし、移動計算機環境においてはこの部分がしばしばボトルネックとなるため、この最適化によって十分な性能向上が期待できる。例えば、エラー率の高い通信メディアを使用しているときは、移動計算機とプロキシエージェント間でローカルな再転送を行い、エンド - エンドのパフォーマンス向上を図る。また、バンド幅の狭い通信メディアを使用しているときには、パケットの圧縮を行なうことでバンド幅の有効な利用を図る。

移動計算機

IETF Mobile IP における移動計算機は、ホームエージェントへの登録や移動の検知、ルーティングテーブルの変更といった機能を持つ。JAIST Mobile IP における移動計算機は、これらの機能に加えて通信メディアを動的に切替える機能、およびプロキシエージェントへの対応としてメディア特性に適應する機能を持つ。通信メディアを切替える際に必要となる情報は移動計算機上の環境サーバから取得し、円滑なメディアの切替えを可能にしている。また、IETF Mobile IP のフォーリンエージェントに相当するものを移動計算機内に搭載することで、プロキシエージェントが存在しないネットワークへの移動も可能である。

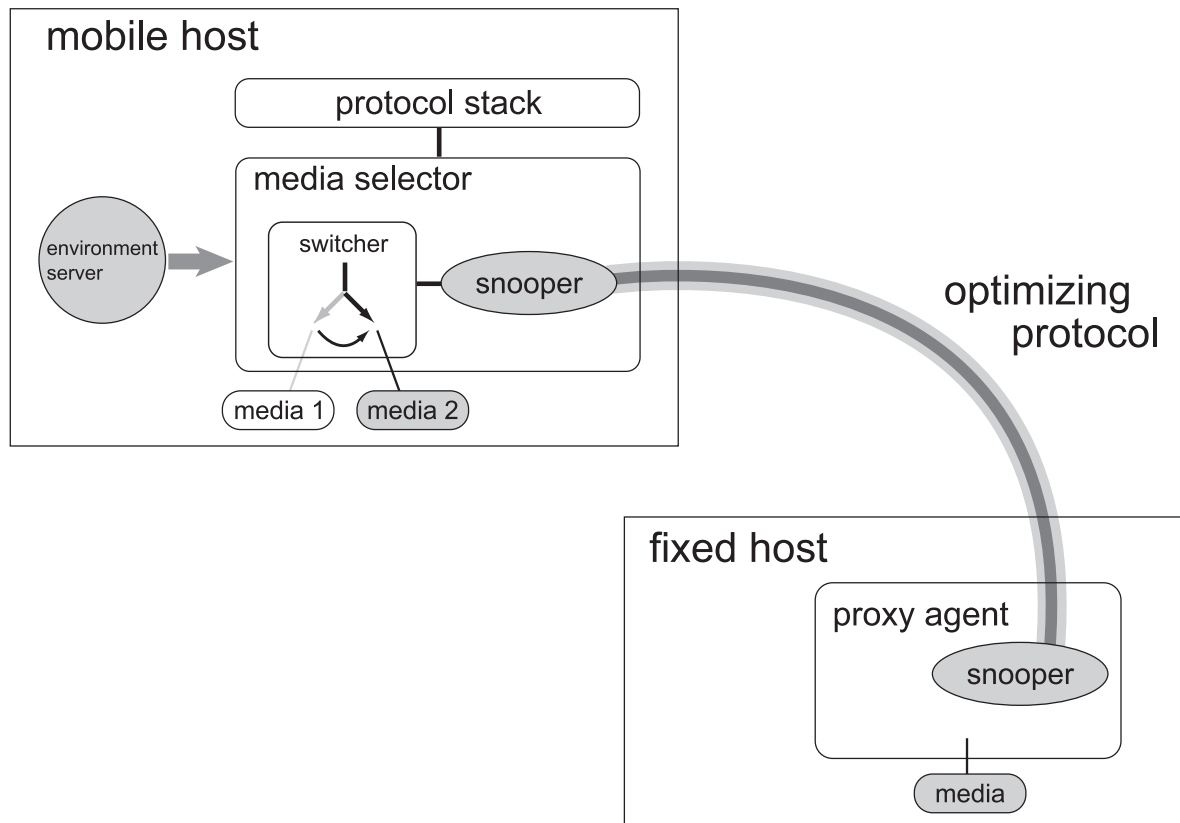


図 4.2: ソフトウェアの構成

移動のためのプロトコル

IETF Mobile IP では移動計算機の登録プロトコル、エージェント発見プロトコル[4] といったいくつかのプロトコルを使用する。JAIST Mobile IP では IETF Mobile IP と互換性を保つためそれらのプロトコルをそのまま使用する。

4.3 ソフトウェアの構成

本節では、JAIST Mobile IP システムを実現するためのソフトウェア構成および、システムの設計・開発方針の概要について述べる。JAIST Mobile IP システムを実現する上で中心となる移動計算機およびプロキシエージェントのソフトウェア構成を図 4.2 に示す。図中の環境サーバ、メディアセレクタ、プロキシエージェントが JAIST Mobile IP システムを構成するために必要となる主なプロセスである。

環境サーバ

環境サーバ [9] は計算機の環境情報をデータベースとして一括管理し、それらを統合化されたインタフェースとともにクライアントアプリケーションに提供するサーバプログラムである。

メディアセクタ

メディアセクタは、移動計算機のネットワークデバイスとプロトコルスタックの間に位置し、最適な通信メディアの選択と自動切換えを可能にするユーザプログラムである。JAIST Mobile IP では、メディアセクタが環境サーバのクライアントアプリケーションとなる。メディアセクタは、環境サーバからネットワークデバイスに関する情報を取得し、それをもとに通信メディアの切替のタイミングを制御する。実際にメディアの切替を行うのはメディアセクタ内のスイッチャモジュールである。スイッチャモジュールは移動計算機が持つ複数のネットワークデバイスをプロトコルスタックに対して仮想的に一つのデバイスとして見せる機能を提供し、これによって通信メディアの切替を容易にしている。また、このような機構を TCP/IP プロトコルスタックの下位に置くことにより、従来のネットワークシステムやアプリケーションをそのまま利用することができる。

プロキシエージェント

プロキシエージェントは従来の Mobile IP システムのフォーリンエージェントに代わるものである。プロキシエージェントを起動するホストはフォーリンネットワーク上の固定ホストであり、従来のフォーリンエージェントの移動サポートに加えて、移動計算機が使用する通信メディアの特性に適応する機能を持つ。メディアセクタとプロキシエージェント内に存在するスヌーパが、この機能を実現するためのモジュールである。

スヌーパ

スヌーパモジュールは、移動計算機とプロキシエージェント間を行き来するパケットを監視しながら対応する処理を行うことで、使用中の通信メディアの特性に合わせた最適化を行う。従来の Mobile IP システムは、移動計算機上のアプリケーションから送信されるパケットには何ら手を加えないが、我々のシステムではこれらのパケットにもカプセル化を施し、双方向のパケットが必ずプロキシエージェントを通過するように拡張を加えた。これによって、移動計算機とプロキシエージェント間を行き来する全ての IP パケットをスヌーパで監視する。このようにパケットを監視するレイヤを IP 層の下位に置くことで、

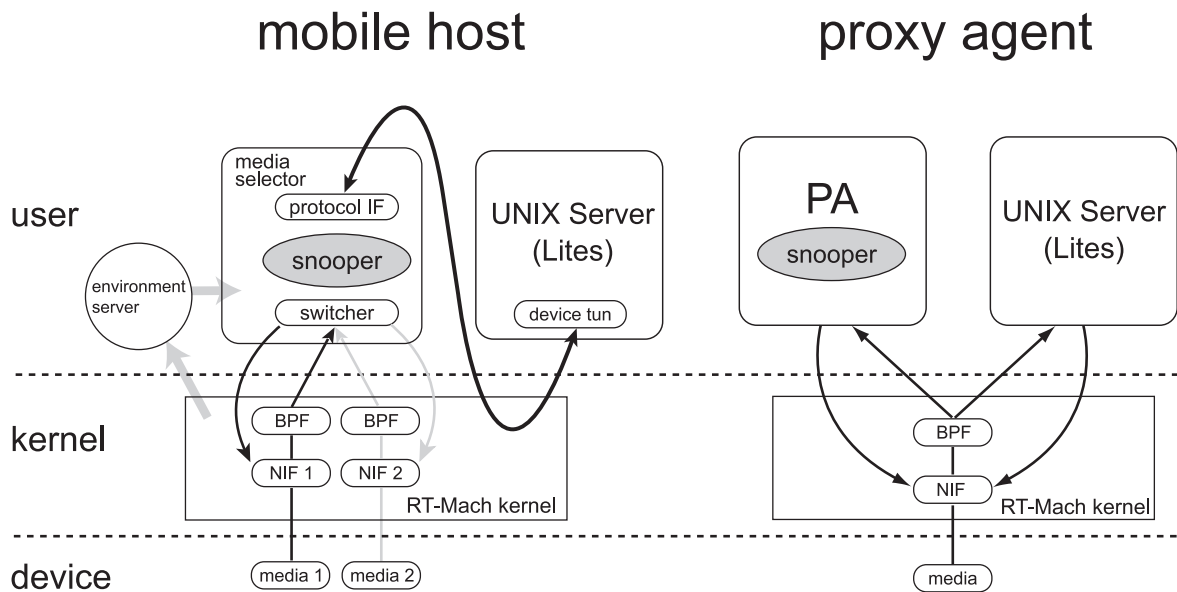


図 4.3: RT-Mach への実装

最適化ポリシーの動的な変更や追加を可能にする。具体的には、通信メディアの特性ごとに最適化のためのモジュールを用意し、特性の変化に応じて使用するモジュールを動的に変更する。

4.4 RT-Mach への実装

図 4.2 のシステム構成をもとに、移動計算機とプロキシエージェントの実装を行った。ベースとして用いているのは、RT-Mach マイクロカーネルと Unix サーバ (Lites) の組み合わせである。OS と各モジュールの位置関係を図 4.3 に示す。

RT-Mach は、Mach に実時間拡張を行ったマイクロカーネルである。マイクロカーネルでは、アドレス空間、スレッド、プロセス間通信、デバイス管理などの基本的な機構のみを提供し、それ以外の OS サービスは、ユーザレベルのアドレス空間にサーバとして実装される。たとえば、UNIX などの単層構造の OS においてカーネルの内部に実装されているファイルサービスやネットワークサービスなどの OS サービスは、ユーザレベルのサーバ内に実装される。Lites は、この OS サービスを提供するサーバで、4.4BSD Lite の機能を提供している。Lites 上では、FreeBSD 2.2 上でコンパイルされたプログラムの多くをそのまま実行することができる。

環境サーバはユーザレベルのプロセスであり、RT-Mach カーネルから得られる計算機環境の様々な情報をアプリケーションに提供できる形で保持している。たとえば、特定のデバイスがユーザによって差し替えられたことを検知すると、それをアプリケーションに通知するといったことが可能である。また、環境サーバに対する問い合わせは明確に定義されており、アプリケーションはシステムに関する様々な情報をシンプルなインタフェースを用いて利用することができる。JAIST Mobile IP システムでは、メディアセクタが環境サーバを利用しネットワークデバイスの情報を効率よく取得することで、変化の激しい環境への適応を実現している。

メディアセクタもまたユーザレベルのプロセスであるが、プロトコル階層的にはIP層の下に位置する。メディアセクタをこの位置に配置することにより、通信メディアの切替に伴うIP アドレスの変化をスイッチャのレベルで隠蔽し、通信メディアの仮想化を実現している。移動計算機の Lites はトンネルデバイスに付けられたアドレスを用いて仮想的にネットワークに接続する。つまり、Lites の TCP/IP プロトコルスタックに対して仮想的な IP アドレスを与えることで、Lites 上のアプリケーションは常にこの IP アドレスを用いてパケットの送受信を行う。Lites がトンネルデバイスに書き込んだパケットはメディアセクタによって取り込まれ、スヌーパ等で適切な処理を行った後、ネットワークインタフェースに渡される。また、ネットワークインターフェースに到着したパケットは、カーネルのパケットフィルタを介してメディアセクタが取り込み、スヌーパ等に適切な処理を行わせた後、トンネルデバイスに書き込む。このように、Lites はメディアセクタの存在を意識せずにトンネルデバイスに対してパケットの入出力を行う。

この実装の利点は、従来のネットワークアプリケーションを変更することなくそのまま利用できることにある。ただし、移動計算機のメディアセクタがユーザ空間にあり、Lites とのパケットの受け渡しにトンネルデバイスを用いていることから相当のオーバーヘッドが予測される。ここで、パケット処理の高速化を図るためにカーネルやプロトコルスタックに手を加える実装も考えられるが、システム全体を通しての拡張性を優先し、この実装方針を採用した。ユーザ空間上に独立したプロセスとして存在するメディアセクタは、通信メディアを選択する際の新しいポリシーや、様々な通信メディアに対応した最適化ポリシーの追加を容易に実現する。

また、PA プロセスは、Mach カーネルのパケットフィルタを用いて処理の対象となるパケットをフィルタリングすることで、Lites とは完全に独立したプロセスとして機能する。たとえば、プロキシエージェントが移動計算機宛てのパケットを受け取ると、そのパケットは Lites には向かわずに、PA プロセスによってパケットスヌーピング等の処理が行われた後、ネットワークインタフェースに戻されて移動計算機に配送される。

現状では、移動計算機が複数利用可能な通信メディアを持つ状況で、メディアセレクタによる通信メディアの自動選択と動的切替えが可能となっている。この際には環境サーバから提供される情報を利用する。サポートする通信メディアは、イーサネット、Netwave、シリアル接続 (PPP) の 3 種類である。移動計算機が接続しているネットワークにプロキシエージェントが存在する場合には、これを利用して使用中の通信メディアに合わせた最適化を行うことが可能である。

第 5 章

通信メディア特性の変化への適応

JAIST Mobile IP システムはパケットスヌープング機構を提供することによって通信メディアの特性の変化への柔軟な適応を実現する。この機構を用いることにより、移動計算機環境において、通信メディアの切り替えやロケーションの変化に伴うメディア特性の変化に適応し、できる限り効率の良いデータ転送を維持することが可能となる。本章では、この機構の特徴および設計と実装について詳述する。

5.1 最適化ポリシーの動的な変更

現在、TCP/IP は通信メディアごとにさまざまな最適化が提案されているが、それらは各通信メディアの特性と密に結びついているため、計算機の移動や通信メディアの動的な変更には対処しきれていない。通信状態の変化の激しい移動計算機環境においては、メディアの特性に特化したそれらの最適化手法を動的に切替えて利用できることが理想的である。

一方、こうした最適化が提案されてきた理由は、移動計算機の用いる通信メディア、たとえば無線や PHS などが、他の固定ホストで用いられているイーサネットなどの有線メディアに比べて、通信性能が劣るためである。つまり、移動計算機とネットワークを接続するための回線が、通信路上のボトルネックとなる場合が多い。とくに、従来の Mobile IP システムでは、移動計算機とフォーリンエージェントを結ぶ回線が、このボトルネックになると考えられる。つまり、移動計算機が外部のネットワークに接続する場合、従来の Mobile IP システムではネットワーク上のフォーリンエージェントと 1 ホップで接続する形をとるため、ここで貧弱なメディアが使われていると移動計算機とフォーリンエージェントを結ぶ回線が上記のボトルネックになってしまう。

こういった背景を踏まえて、JAIST Mobile IP システムでは、ネットワーク上のフォーリンエージェントの位置に着目し、これをより積極的に利用する方法を提案した。本来、IETF Mobile IP が規定するフォーリンエージェントは、計算機の移動を可能にするためだけに存在するが、JAIST Mobile IP では、フォーリンエージェントを基盤とし、通信メディアの特性の変化に適応するための機能を持つプロキシエージェントを提供する。プロキシエージェントと移動計算機が協調して動作することによって、ボトルネックの解消を図る。また、移動計算機は移動先のネットワーク上のプロキシエージェントと協調し、使用している通信メディアの特性に応じてパケットの送信を最適化する。具体的には、移動計算機とプロキシエージェントの間で独自のプロトコルを用いることにより、様々な最適化の手法を通信メディアの特性に応じて使い分ける。例えば、エラー率の高い通信メディアを使用しているときは、移動計算機とプロキシエージェント間でローカルな再転送を行い、エンド - エンドのパフォーマンス向上を図る。また、バンド幅の狭い通信メディアを使用しているときには、パケットの圧縮を行なうことでバンド幅の有効な利用を図る。

5.2 移動計算機 - プロキシエージェント間プロトコル

IETF Mobile IP では、ホームエージェントからフォーリンエージェントに向かうパケットに IPinIP のカプセル化を施す。フォーリンエージェントの役割は主にこのカプセル化を外し、通常のパケットに戻して移動計算機に届けることである。従来のシステムでは、フォーリンエージェントはこのカプセル化の処理を IP よりも下位の層で行っている。そこで、このカプセル化の処理を行う部分に、最適化の処理を新たに追加することで特性変化への適応、すなわち最適化ポリシーの動的な変更を可能にする。この処理がオーバーヘッドになるような通信メディア、たとえばイーサネットなどをを用いるときには、プロキシエージェントを即座に切り離すことができる。移動計算機は、そのためにフォーリンエージェントをエミュレーションする機能を持っている。

また、従来の Mobile IP システムでは、移動計算機から送信されるパケットはフォーリンエージェントを介さないが(図 5.1)、JAIST Mobile IP ではこのパケットにもカプセル化を施し、双方向のパケットが必ずプロキシエージェントを通過するように拡張を加えた(図 5.2)。移動計算機とプロキシエージェントの間を流れるカプセル化パケットのヘッダは、IP ヘッダを若干変形させたものである(図 5.3)。

ヘッダ中の OPT TYPE フィールドが現在用いている最適化ポリシーを示す。最適化ポリシーの決定は移動計算機主導で行う。プロキシエージェントは移動計算機から送られてきた

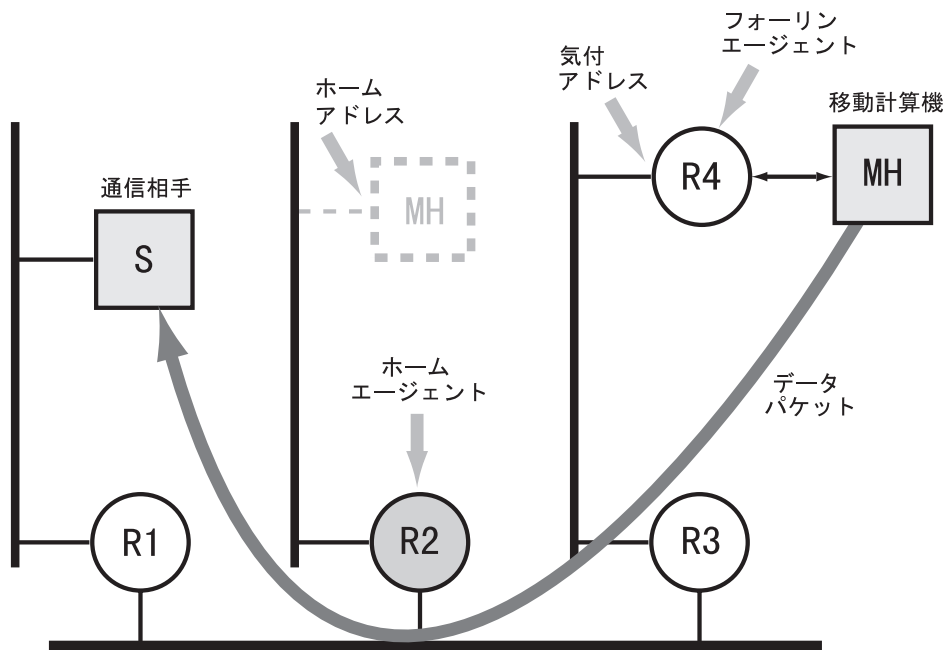


図 5.1: IETF Mobile IP におけるパケットの流れ

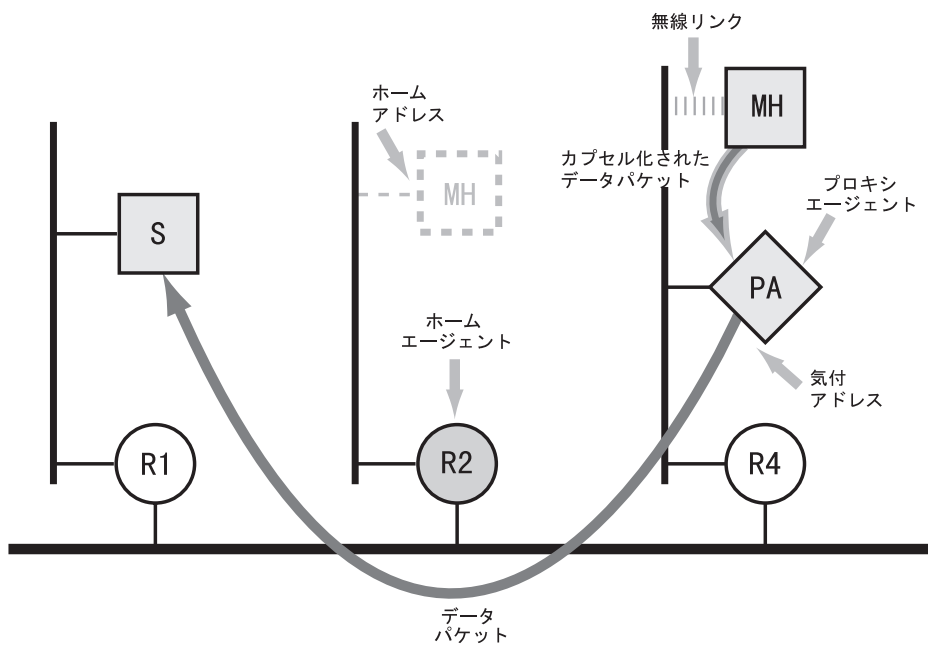


図 5.2: JAIST Mobile IP におけるパケットの流れ

0	4	8	16	31
V = 4		HL=5	OPT TYPE	TOTAL LENGTH
IDENTIFICATION			FRAG OFFSET = 0	
TTL = 1		PROT=RAW IP	HEADER CHECKSUM	
SOURCE IP ADDRESS				
DESTINATION IP ADDRESS				
DATA				
...				

図 5.3: Snoop ヘッダ

パケットのヘッダの OPT TYPE フィールドを常に監視しており、移動計算機が現在どの最適化を使おうとしているかを判断する。移動計算機とプロキシの間は通常 1 ホップであるので、TTL フィールドは通常 1 である[†]。また、プロトコルフィールドには RAW IP を設定する。このフィールドを用いて、プロキシエージェントは最適化の処理を行うべきパケットを判断する。移動計算機からプロキシエージェントに向かうパケットのソースアドレスフィールドには、移動計算機がそのネットワークにおいて DHCP 等で獲得したアドレスが入る。デスティネーションアドレスはプロキシエージェントのアドレスである。移動計算機からプロキシエージェントに向かうパケットの各アドレスフィールドは、ソースとデスティネーションが逆になる。すなわちサブネット上の他のマシンから見ると、移動計算機とプロキシエージェント間を流れるカプセル化パケットは通常の IP パケットと同様に見える。

最適化モジュールによってはヘッダの後にオプションとして、モジュールどうしが交換する情報が含まれる場合もある。たとえば、エラーリカバリの選択的確認応答などがこれに相当する。データ部は、基本的に、移動計算機が通信相手とやり取りする IP パケットである。また、被カプセル化パケット全体を圧縮したり、暗号化したりする場合は、バイナリデータとなる。

[†]PPP サーバなどを経由する場合は、この限りではない。

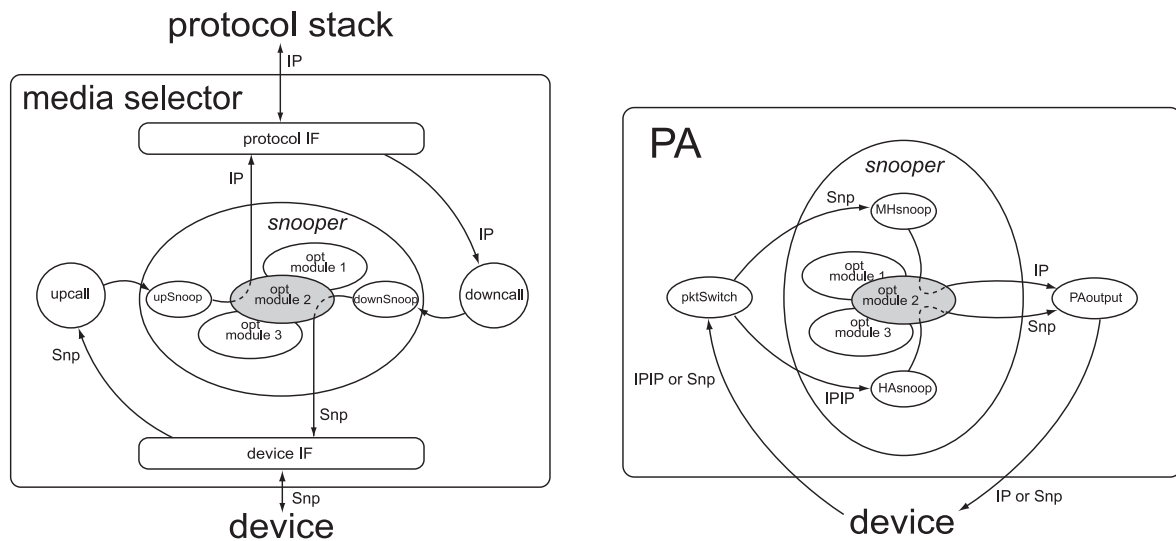


図 5.4: スヌーパ

5.3 パケットスヌーパ

最適化ポリシーの動的な変更は、移動計算機とプロキシエージェントにパケットスヌーパを設けることで実現する (図 5.4)。

前述したように、JAIST Mobile IP では双方向の packets が必ずプロキシエージェントを通過するように拡張を加えた。これによって、移動計算機とプロキシエージェント間を行き来する全ての IP packets をスヌーパで監視する。スヌーパは移動計算機内のメディアセレクタと連動しており、現在使用している通信メディアの特性に合わせた最適化処理を行う。カプセル化された packets がスヌーパ内を通過する際に、状況に応じて適切なモジュールに渡すことで、メディアの特性を活かした最適化を可能にする。最適化モジュールは統一されたインタフェースを持ち、新しいモジュールの追加、変更は容易である。以下では、スヌーパの packets 処理の詳細について述べる。

5.3.1 メディアセレクタ内のスヌーパ

メディアセレクタはアップコールとダウンコールという二つのスレッドを持ち、これら二つのスレッドが、Lites とデバイス間を行き来する packets を適切なタイミングで取り込んでいる。アップコールはネットワークデバイスに到着する packets を監視しており、それを取り込んで最終的に Lites のプロトコルスタックに届ける役割を果たす。ダウ

ンコールは Lites がプロトコルインタフェースに書き込むパケットを監視しており、それを取り込んで最終的にネットワークデバイスに出力する役割を果たす。スヌーパを用いた最適化を行う場合、アップコール、ダウンコールは、それぞれスヌーパのインタフェースに対して取り込んだパケットを渡す。スヌーパのインタフェースはアップとダウンで別々に用意されており、それぞれ、図 5.4 の upSnoop、downSnoop という手続きがこれに相当する。

手続き upSnoop は、アップコールから受け取った Snp 形式のパケットのヘッダの OPT TYPE フィールドをチェックし、これをもとにどの最適化モジュールを使うべきかを判断する。これと同時にヘッダを取り除き、データ部分のみを最適化モジュールに渡す。最適化モジュールに渡されるデータは、そのモジュールが規定する形式にしたがったものである。たとえば、圧縮モジュールを用いている場合には、利用している圧縮アルゴリズム等の情報が含まれており、最終的にプロトコルインタフェースに出力する段階では、送信元が出力した IP パケットに戻される。

手続き downSnoop は、ダウンコールから受け取った IP パケットに図 5.3 のヘッダを付け加える。このとき、メディアセレクタが管理する情報から用いるべき最適化を決定し、OPT TYPE フィールドにこれを設定したあと、対応する最適化モジュールにパケットを渡す。最適化モジュールは、ヘッダ以下の部分(もとの IP パケット)に必要な処理、たとえば圧縮や暗号化を施し、デバイスインタフェースに出力する。

5.3.2 プロキシエージェント内のスヌーパ

PA プロセス内の手続き pktSwitch が、パケットフィルタを用いて必要なパケットを取り込み、移動計算機からのパケットとホームエージェントからのパケットを対応する手続きに振り分けている。振り分けの判断は取り込んだパケットのプロトコルタイプフィールドをチェックすることで行う。ホームエージェントから届くパケットは IPinIP であり、移動計算機から届くパケットは、RAW IP となっている。

移動計算機からのパケットは手続き MHsnoop に渡され、ヘッダの OPT TYPE フィールドを調べてどの最適化モジュールを使うべきかを判断する。同時に移動計算機を管理するテーブルを調べ、それまでとは異なる最適化モジュールを指定してきている場合には、この情報を更新する。次に、MHsnoop はヘッダを取り除いたデータを対応する最適化モジュールに渡す。最適化モジュールはこのデータをもとの IP パケット(Lites 上のアプリケーションが送信したもの)に戻して、手続き PAoutput に渡す。PAoutput はこの IP パケットをネットワークデバイスに出力する。プロキシを離れた後、この IP パケットは通

常の IP ルーティングによって移動計算機の通信相手に届けられる。

ホームエージェントからのパケットは、手続き HAsnoop に渡される。HAsnoop が受け取るパケットは本来、Mobile IP のフォーリンエージェントが受け取る IPinIP カプセル化パケットである。HAsnoop はこのパケットに対して再カプセル化を行う。このとき、できる限り無駄を省くために、IPinIP のヘッダをほとんどそのまま利用し、変更が必要なフィールドだけを書き換える。同時に上記の移動計算機管理テーブルを引いて、OPT TYPE フィールドに最適化のタイプをセットする。再カプセル化されたパケットは、最適化モジュールに渡され、ヘッダ以下の部分に対応する処理が施される。このパケットは最終的に PAoutput に渡されて、デバイスに出力され、移動計算機に届けられる。

5.3.3 最適化モジュールのインターフェース

新しい最適化モジュールの追加や、既に組み込まれているモジュールの変更を容易にするために、パケットスヌーパは、最適化モジュールに対して共通のインターフェースを提供している。各モジュールに対して所定の項目の設定を行うことで、スヌーパの振舞いを定義する。

メディアセクタ内のスヌーパの設定

まず、各モジュールは、そのモジュールがどのような状況で使われるべきかを指定する必要がある。現在は、通信メディアの種類、バンド幅、エラー率を考慮している。また、各モジュールは、自分が受け取りたいパケットの種類を指定する。現在は IP のプロトコル番号のみを対象にしている。以上の指定は、スヌーパ内の手続き、upSnoop、downSnoop に反映される。これら 2 つの手続きは、設定された内容をもとに使用するモジュールの判断や、モジュールに渡すパケットを選定する。

一方、各モジュールは、upSnoop、downSnoop に対応してパケットの入り口と出口をそれぞれ用意しなければならない。upSnoop から受け取るパケットは、移動計算機の Lites 上で起動されているアプリケーションに届けられなければならないので、通常の IP パケットとしてメディアセクタのプロトコルインターフェースに出力する。また、downSnoop から受け取るパケットは、デバイスインターフェースに出力する。

PA プロセス内のスヌーパの設定

PA プロセス内のスヌーパの設定には、モジュールを選択するための指定は必要ない。使用するモジュールは移動計算機からのパケットのヘッダの情報を元に切替える。各モ

ジュールが、自分が受け取りたいパケットの種類を指定するのは、メディアセクタ内のスヌーパの場合と同様である。また、この指定が、スヌーパ内の手続き、MHsnoop、HAsnoop に反映されるのも同様である。これら2つの手続きは、設定された内容をもとに使用するモジュールの判断や、モジュールに渡すパケットを選定する。各モジュールが、MHsnoop、HAsnoop に対応してパケットの入り口と出口をそれぞれ用意しなければならないのは、メディアセクタ内のスヌーパの場合と同様である。

5.4 最適化モジュール

最適化モジュールとしては、現在、パケット圧縮モジュールとエラーリカバリモジュールが利用可能である。以下では、各モジュールの用途と機能の詳細について述べる。

5.4.1 パケット圧縮モジュール

パケット圧縮モジュールは、PHS 等、バンド幅の狭い回線に接続している場合に、移動計算機とプロキシエージェント間の双方向のパケットに圧縮を施し、回線上を通るデータのサイズをできる限り小さくすることで、バンド幅を有効に利用できるようにする。このモジュールは、圧縮処理によるタイムロスが生じてもバンド幅を十分に使い切ることができるような状況で使用した場合に効果がある。

移動計算機のメディアセクタが圧縮モジュールを使うと判断した場合、移動計算機とプロキシエージェント間を流れるカプセル化パケットは、5.5のようになる。実際に圧縮を施すのは、元の IP パケット全体であり、圧縮後のデータに図 5.3 のヘッダを付け加える。圧縮モジュールを利用しているときの移動計算機およびプロキシエージェント内のスヌーパのパケット処理を図 5.6 に示す。

移動計算機側のモジュール

圧縮モジュールを用いているとき、移動計算機のネットワークデバイスが受け取るパケットは、図 5.3 のヘッダに続いて圧縮データが入っている (Snp(data) 形式)。アップコールスレッドは、常にデバイスインタフェースを監視しており、パケットが到着するとそれを取り込んで、upSnoop に渡す。この時点まで、Snp(data) のパケット形式に変わりはない。ここで、upSnoop はカプセル化を外し、圧縮モジュール内の手続き uncomp に圧縮データ (data) を渡す。uncomp が圧縮データを解凍すると、送信元が送り出したときの IP パケットになる。これを Lites に届けるためにプロトコルインタフェースに出力する。

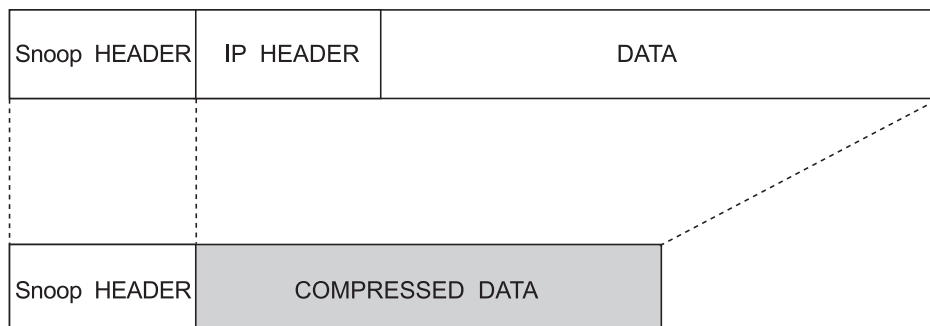


図 5.5: 圧縮パケット

また、Lites によってプロトコルインタフェースに書かれた IP パケットは、ダウンコールスレッドによって取り込まれ、続いて downSnoop に渡される。downSnoop はダウンコールから受け取った IP パケットに図 5.3 のヘッダを付け加えたパケット (Snp(IP) 形式) を圧縮モジュール内の手続き comp に渡す。comp は Snp ヘッダに続く IP パケットに圧縮を施し、デバイスインタフェースに対して Snp(data) の形式で出力する。

プロキシエージェント側のモジュール

プロキシエージェントのネットワークデバイスに到着するパケットのうち、PA プロセスが処理しなければならないのは、ホームエージェントから届く IPinIP カプセル化パケット (IP (IP) 形式) と移動計算機から届く圧縮パケット (Snp(data) 形式) である。PA プロセス内の pktSwitch がこれらのパケットを取り込み、同時に振り分けを行う。

Snp(data) パケットは MHsnoop に渡され、MHsnoop はヘッダを取り除いた圧縮データ (data) を圧縮モジュール内の手続き uncomp に渡す。uncomp によって解凍されたデータは移動計算機上のアプリケーションが送信した IP パケットになる。これを PAoutput を用いてデバイスに出力する。一方、IP(IP) パケットは HAsnoop に渡され、HAsnoop はもとの IPIP ヘッダを Snp ヘッダに変更して、圧縮モジュール内の手続き comp に渡す。comp は Snp に続く IP パケットを圧縮し、Snp(data) パケットとして最終的に PAoutput を用いてデバイスに出力する。

このように、移動計算機とプロキシエージェントの圧縮モジュールは、対称になっており、内部的には圧縮部 comp と解凍部 uncomp に分かれている。現在、圧縮のエンジンとしては、LZ78 を用いているが、アプリケーションのヘッダ情報などからデータの特性が予測できるときには、有効なアルゴリズムを用いた圧縮エンジンを使い分けるといった拡

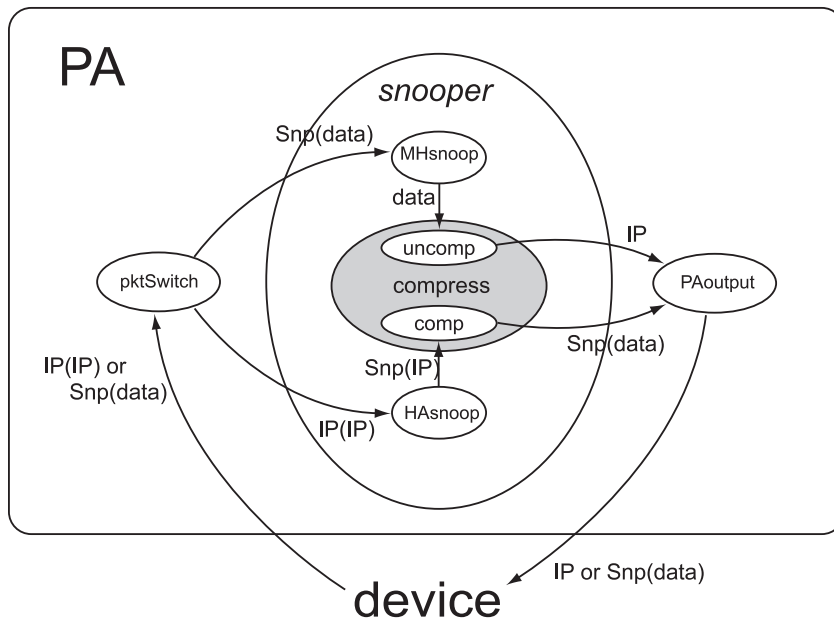
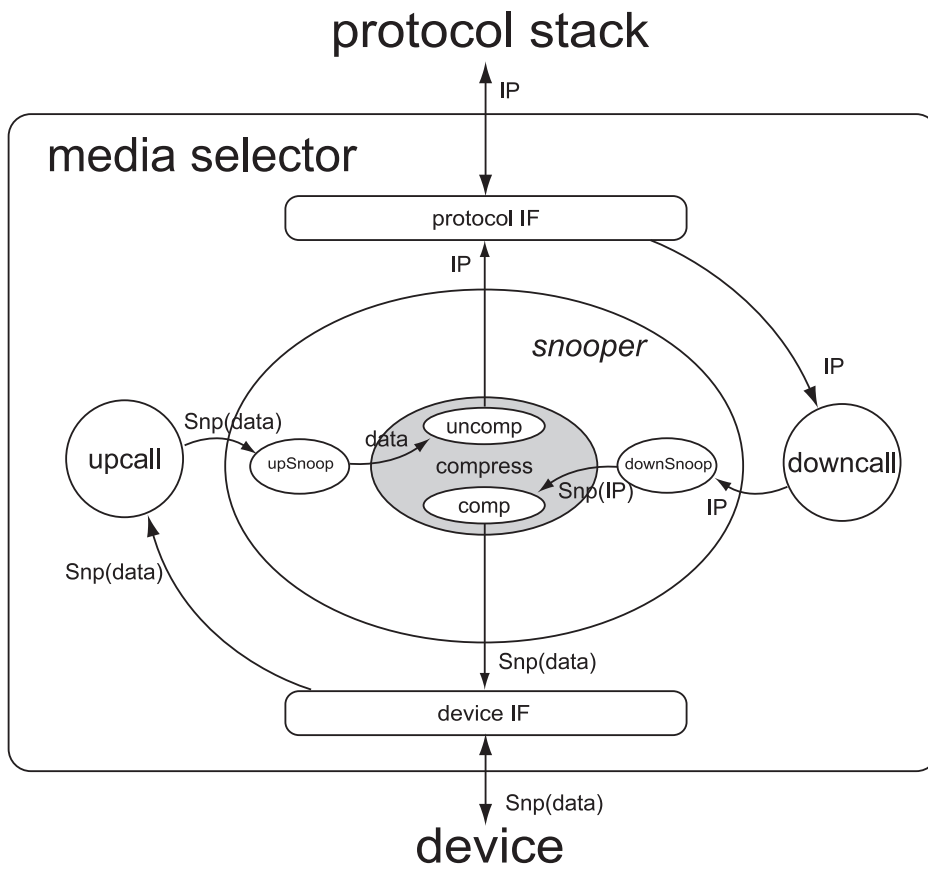


図 5.6: パケット圧縮モジュール

張が考えられる。

5.4.2 エラーリカバリモジュール

エラーリカバリモジュールは、無線 LAN 等、エラー率の高い回線に接続している場合に、TCP のデータ転送において高速なエラー回復を可能にするため、移動計算機とプロキシエージェント間でローカルな再転送を処理する。

従来の TCP はパケット損失の主原因が輻輳にあるような有線ネットワークで十分に機能するように最適化されている。一方、近年登場してきた無線リンクにはビットエラー率が有線の場合よりも高くなるという特徴があり、無線リンクを利用したネットワークのパケット損失は、多くの場合、このビットエラーによって生じる。このような特徴を持つネットワークで従来の TCP をそのまま用いると、たとえば輻輳回避や高速リカバリのアルゴリズムが不必要に働いてスループットを低減させてしまう可能性がある。そこで、メディアセクタと PA プロセス内のスヌーパにエラーリカバリモジュールを配置し (図 5.7)、移動計算機とプロキシエージェント間でローカルな再転送を可能にする。つまり、移動計算機とプロキシエージェント間で生じたパケットの損失を TCP に対してできる限り隠し、従来の TCP が持つ最適化メカニズムが不必要に起動されるのを防ぐ。

基本的には TCP のデータパケットをキャッシングし、確認応答パケットを監視することで失われたパケットのローカルな再転送を可能にする。

プロキシエージェントから移動計算機に向かうデータのリカバリ

プロキシエージェントから移動計算機に向かうデータのエラーリカバリには、プロキシエージェント側のリカバリモジュールに、移動計算機から確認応答が届いていない TCP データパケットをキャッシングする機能 (cacheData) と、移動計算機からの TCP 確認応答を監視する機能 (snoopAck) を持たせることで対処する。

cacheData は、移動計算機に向かう TCP データパケットのコピーを各コネクションごとに対応するバッファに格納し、パケットの損失を検出した場合には、このバッファの中からパケットの再転送を行う。バッファ内のパケットのコピーはシーケンス番号順にソートされており、再転送パケットを検索する処理を容易にしている。

snoopAck がリカバリメカニズムの中心である。図 5.8 に手続き snoopAck の流れを示す。snoopAck はまず、移動計算機から届いた確認応答が、それまでのものよりも新しいものであるかどうかをチェックする。新しいものであれば、その確認応答が示す TCP のパケットをリカバリモジュールのパケットバッファからクリアし、ラウンドトリップタイ

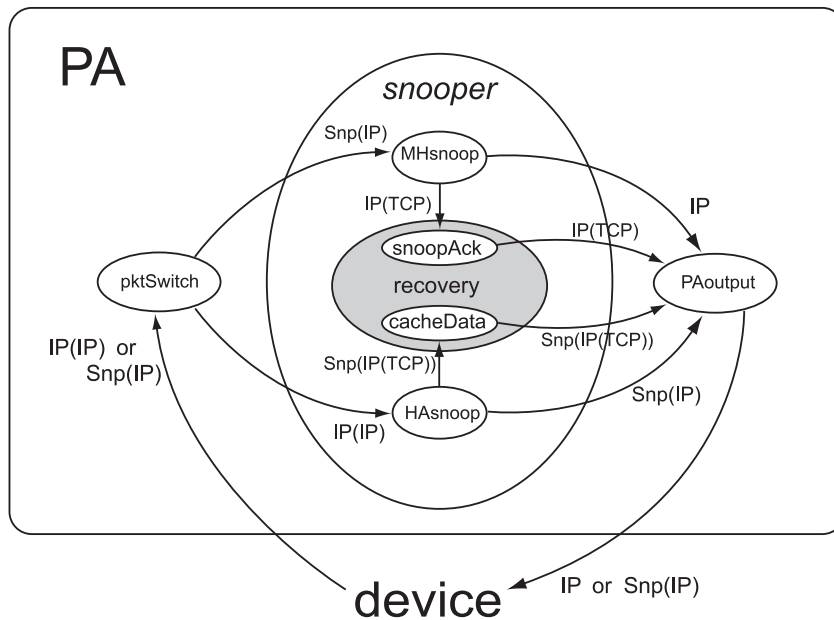
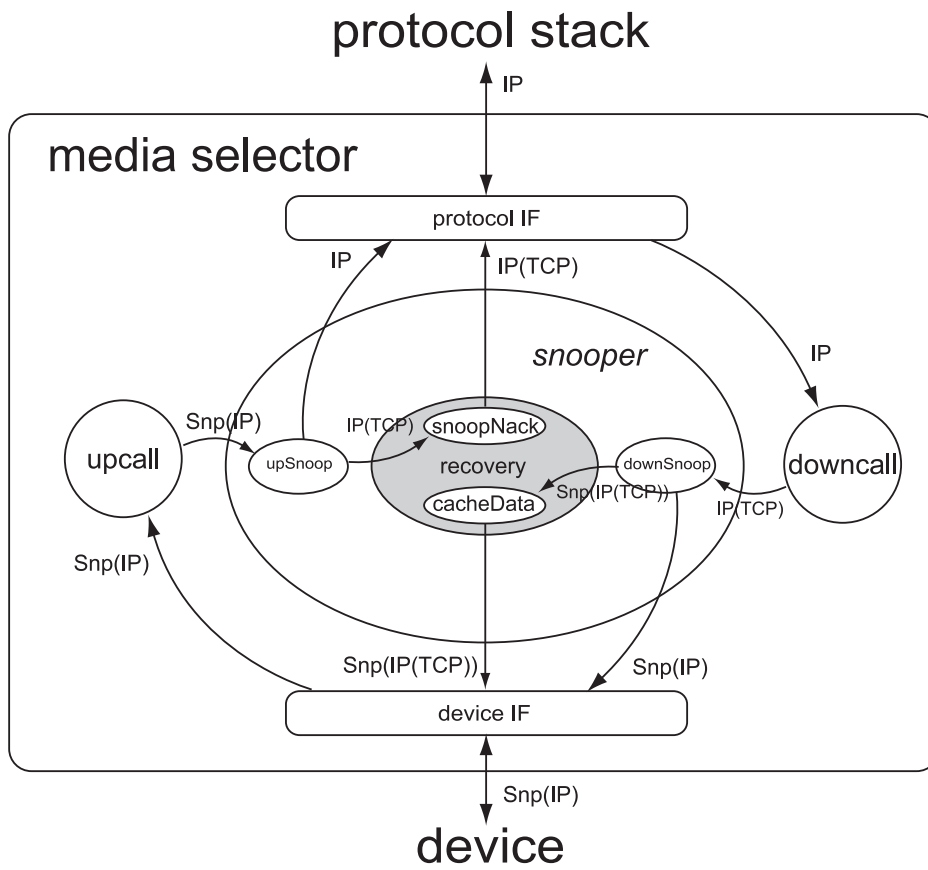


図 5.7: エラーリカバリモジュール

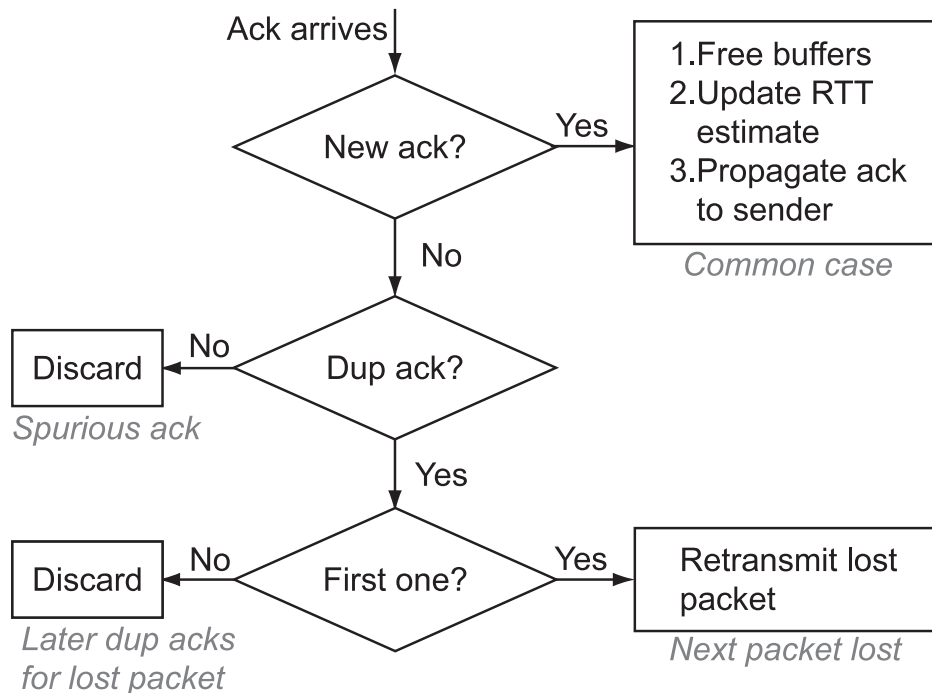


図 5.8: PA スヌーパ内の snoopAck の流れ

ムを計算し直して(ただし、これは、すべての確認応答に対して行うのではなく、基本的に1ウィンドウに1つのパケットの割合で行う)、その確認応答を通信相手に伝達する。これが最も一般的なケースである。ラウンドトリップタイムの計算は、確認応答が返ってくるまでの時間を予測するために行われる。つまり、予測された時間内に移動計算機から確認応答がなければ、パケットの損失が発生したものとみなして、パケットバッファから対応するパケットを再送する。

新しい確認応答でなかった場合は次に、その確認応答が重複確認応答であるかどうかをチェックする。重複確認応答は、順番が狂って届いたパケットに対してTCPが返すものであり、それまでに応答している最大のシーケンス番号を返す。ここで、重複確認応答ではなかった場合、つまり、最大の確認応答番号よりも小さいシーケンスを応答するものであった場合は、snoopAckは論理的におかしいものとしてその確認応答を捨てる。

重複確認応答であった場合、snoopAckはその重複確認応答がはじめて届いたものであるかどうかをチェックする。はじめて届いたものであった場合、パケットバッファの中から対応するパケットを取り出して再転送を行う。2回目以降のものは、損失したパケットに続いて届けられた一連のパケットに対して返されたものであるから無視する。

また、重複確認応答は、いずれの場合であっても送信相手には伝達しない。これによっ

て、通信相手の TCP に対してパケット損失の発生を隠し、不必要な輻輳コントロールを防ぐ。

移動計算機からプロキシエージェントに向かうパケットのリカバリ

移動計算機からプロキシエージェントに向かうデータパケットのエラーリカバリには、プロキシエージェントの `snoopAck` が TCP データのシーケンス番号を監視することで対処する。パケットの損失を検出した場合、移動計算機のリカバリモジュールに対して選択的確認応答を送り、対応するパケットを再転送させる。

したがって、移動計算機のリカバリモジュールは、確認応答されていないデータパケットをキャッシュしておく必要があり、プロキシエージェントのリカバリモジュールからの選択的確認応答に答えることができないなければならない。移動計算機のリカバリモジュールは、これを行う手続きとして `cacheData` と `snoopNack` を持つ。`cacheData` は、Lites が送り出した TCP データパケットをコピーし、バッファにキャッシュする。`snoopNack` は、移動計算機に届く TCP の確認応答とプロキシエージェントのリカバリモジュールから送られてくる選択的確認応答を監視する。`snoopNack` が通常の確認応答を検出した場合、対応するパケットをバッファ内からクリアする。また、選択的確認応答を検出した場合は、対応するパケットをバッファ内から探し出し、見つければそれを再送する。

以上のメカニズムを用いて、Lites の TCP に対して、パケット損失の発生を隠すことにより、輻輳回避などが不必要に発生することを防ぐ。

第 6 章

評価と考察

本章では、測定の結果をもとにパケットスヌーピング機構の効果を示す。測定を行うために、パケットスヌーピング機構を組み込んだ JAIST Mobile IP システムを稼動させる環境を構築した。この実験システムは第 4 章で述べた JAIST Mobile IP システムの機能をほぼ含んでいる。さらに、この実験システムの動作例をもとに考察を行い、パケットスヌーピング機構および JAIST Mobile IP システムの有用性について議論する。

6.1 最適化モジュールの評価

パケットスヌーピングによる最適化の効果を示すために、パケット圧縮モジュール、エラーリカバリモジュールを使用した場合のデータ転送のパフォーマンスを測定した。測定に用いた環境を図 6.1 に示す。

本測定において、移動計算機を除くエージェントおよびルータは、10BASE-T のイーサネットを用いて接続している。移動計算機は、PC カードインタフェースに無線 LAN 用の PC カードを挿入し、ネットワークに接続した。無線 LAN には、Netwave を使用しており、仕様上のバンド幅は上限 1 Mバイトである。ホームエージェントと送信元の PC の OS には、FreeBSD 2.2.7 を用い、プロキシエージェントと移動計算機では、Real-Time Mach と Lites をベースとし、本論文にて提案、実装した JAIST Mobile IP システムが稼動している。

6.1.1 パケット圧縮モジュールの効果

本節では、パケットの圧縮がどの程度有効であるのかを検証するために、圧縮率、圧縮処理のオーバーヘッド、および通信メディアのバンド幅との関係を実際の測定をもとに考

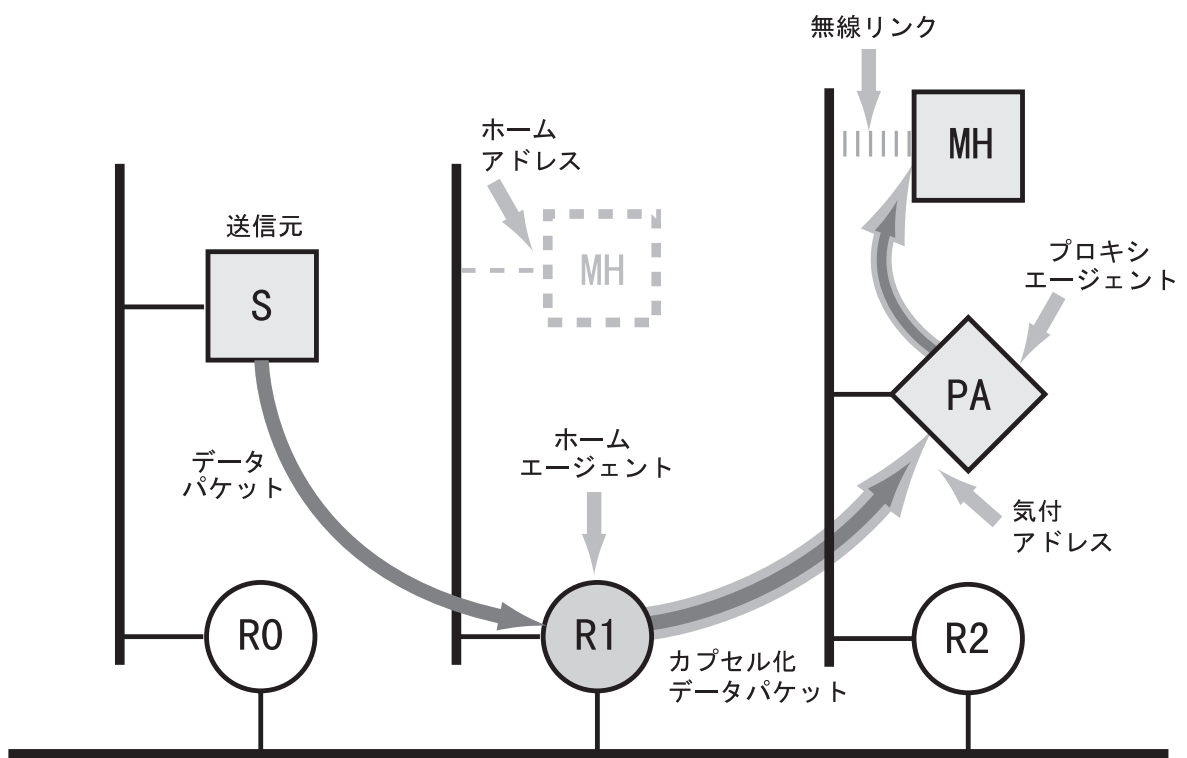


図 6.1: 測定環境

察する。

圧縮モジュールのパフォーマンスは、処理の対象となるデータの圧縮率によって異なる。そこで、本測定では、サイズがほぼ等しい3種類のファイル(テキストデータファイル、バイナリデータファイル、圧縮済みデータファイル)を用意し、それぞれのファイルをFTPによって転送した場合に要する時間を計測した。各ファイルのサイズは約1Mバイトである。なお、本測定において、移動計算機と無線の基地局は至近距離にあり、パケットの損失はほとんど発生しない。フォーリンネットワーク上にある移動計算機MHでFTPを実行し、計算機(S)からファイルの転送を行った結果を以下に示す。

ファイルの種類	転送に要した時間(秒)	
	パケット圧縮あり	パケット圧縮なし
テキストファイル	17.1(圧縮率 38%)	26.2
バイナリファイル	23.7(圧縮率 75%)	
圧縮済みファイル	29.5(圧縮率 100%)	

表 6.1: 圧縮モジュールの効果

表中の圧縮率は、圧縮後のパケットサイズの元のパケットサイズに対する割合の平均である。FTPによるファイル転送の測定はあくまで参考データであるが、結果を見る限り、無線LAN程度のバンド幅でパケット圧縮を利用する場合、圧縮率が80%を越えたあたりで、圧縮アルゴリズムのオーバーヘッドによって、パフォーマンスが悪化することが予測できる。

そこで、パケット圧縮モジュールを利用する条件を考察する。通信メディアのバンド幅を a 、サイズ S_1 のデータ転送に要する時間を t_1 とすると、

$$t_1 = \frac{S_1}{a}$$

となる。また、計算機の単位時間当たりの圧縮処理能力を b とすると、サイズ S_1 のデータの圧縮に要する時間 t_c は、

$$t_c = \frac{S_1}{b}$$

で表わされる。圧縮後のデータサイズを S_2 とおくと、

$$S_2 = cS_1$$

であり、 c は特定の圧縮アルゴリズムの圧縮率である。また、 S_2 の転送に要する時間を t_2 とおくと、

$$t_2 = \frac{S_2}{a}$$

である。ここで、圧縮に要する時間 t_c と圧縮されたデータの転送時間 t_2 との合計が、元のデータをそのまま転送する場合の時間 t_1 よりも小さくなるようであれば、圧縮モジュールの使用は有効であると言える。したがって、

$$t_c + t_2 < t_1$$

を満足すればよい。

$$a > 0, b > 0, 0 < c < 1$$

の条件の下でこの式を整理すると、

$$\frac{S_1}{b} + \frac{S_2}{a} < \frac{S_1}{a}$$

$$\frac{S_1}{b} + \frac{cS_1}{a} < \frac{S_1}{a}$$

$$\frac{1}{b} + \frac{c}{a} < \frac{1}{a}$$

$$\frac{a}{b} < 1 - c$$

$$a < b(1 - c) \tag{6.1}$$

となる。

実際に、上記の測定を例にとると、図 6.1 のネットワークにおける最大サイズの IP パケット (1400 バイト) の圧縮に要する時間は、6 ミリ秒程度であり[†]、これを圧縮の処理速度に換算すると 233 Kバイト/秒である。また、TCP を用いたデータ転送における Netwave の実効バンド幅は、40K バイト程度であり、これらの数値を式 6.1 に当てはめると、

$$40 < 233(1 - c)$$

$$\frac{40}{233} < 1 - c$$

$$c < 1 - 0.17$$

$$c < 0.83$$

[†]ペンティアムプロセッサのタイムスタンプカウンタを用いて測定。

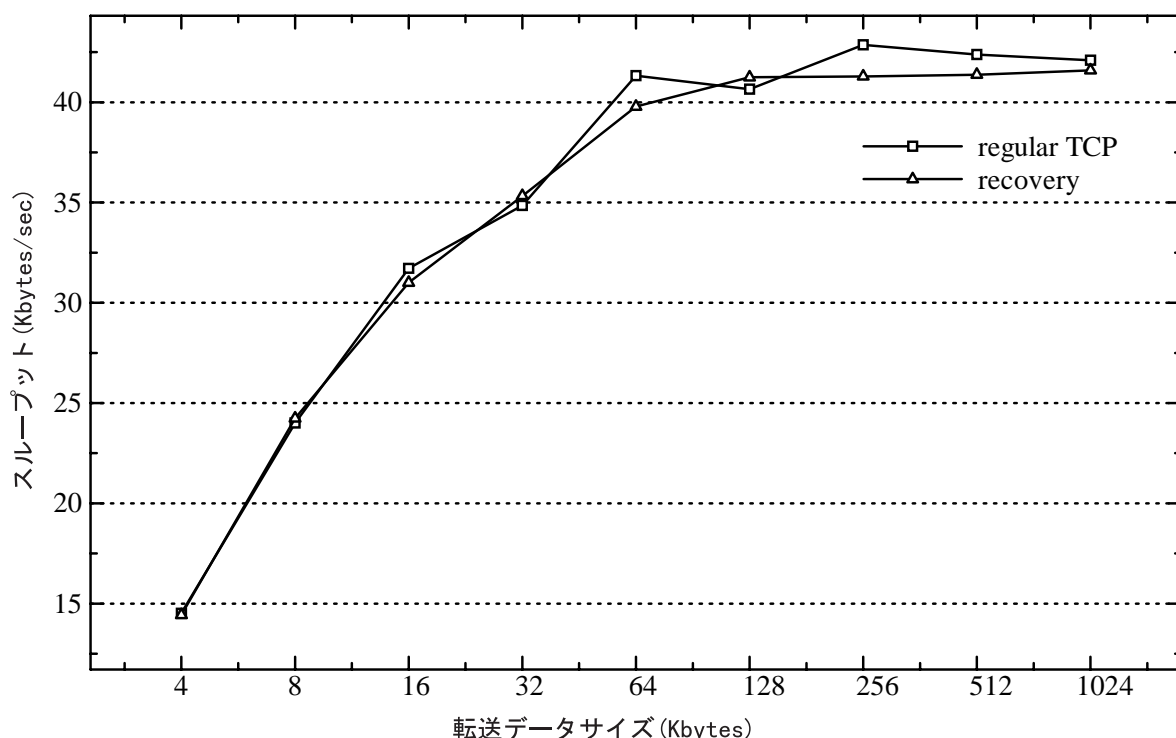


図 6.2: パケットの損失が無視できる場合のスループットの推移

となる。したがって、図 6.1 のシステムにおいては、圧縮率 83% が圧縮モジュールを使用すべきかどうかの判断の参考になる。逆に、いくつかのパケットをサンプリングすることで、ある程度圧縮率の予想が付くならば、それを元に、実効バンド幅に対して圧縮モジュールが有効であるかどうかを判断することもできる。こういった拡張機能の組み込みは、パケットスヌーピング機構の枠組みの中で、比較的容易に実現可能である。

6.1.2 エラーリカバリモジュールの効果

エラーリカバリモジュールはエラー率の高い通信メディアを使用している場合に選択される。そこで、図 6.1 の環境において、移動計算機がネットワークに接続するメディアのエラー率を任意に設定できるように、移動計算機に測定用の機能を追加した。実際には、移動計算機のスヌーパにパラメータとして与えた確率で、パケットをランダムに落とす機能を付け加えている。測定用のベンチマーク・プログラムには `ttcp††` を用い、計算機 S から移動計算機 MH に対してデータの転送を実行した。

^{††} 2 台のシステム間における TCP と UDP の性能を測定するベンチマーク・ツール。米陸軍弾道学研究所 (BRL) で開発され、パブリック・ドメインとして提供されている。

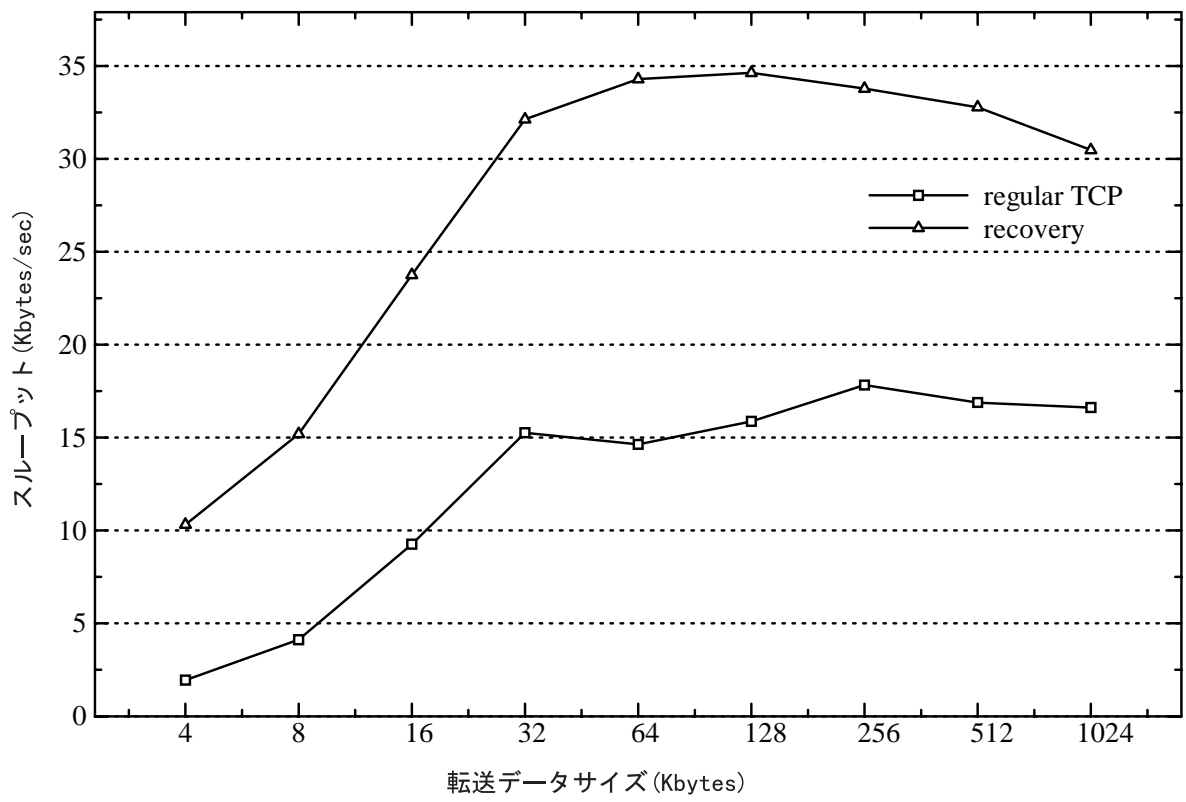


図 6.3: パケット損失率 5 %時のスループットの推移

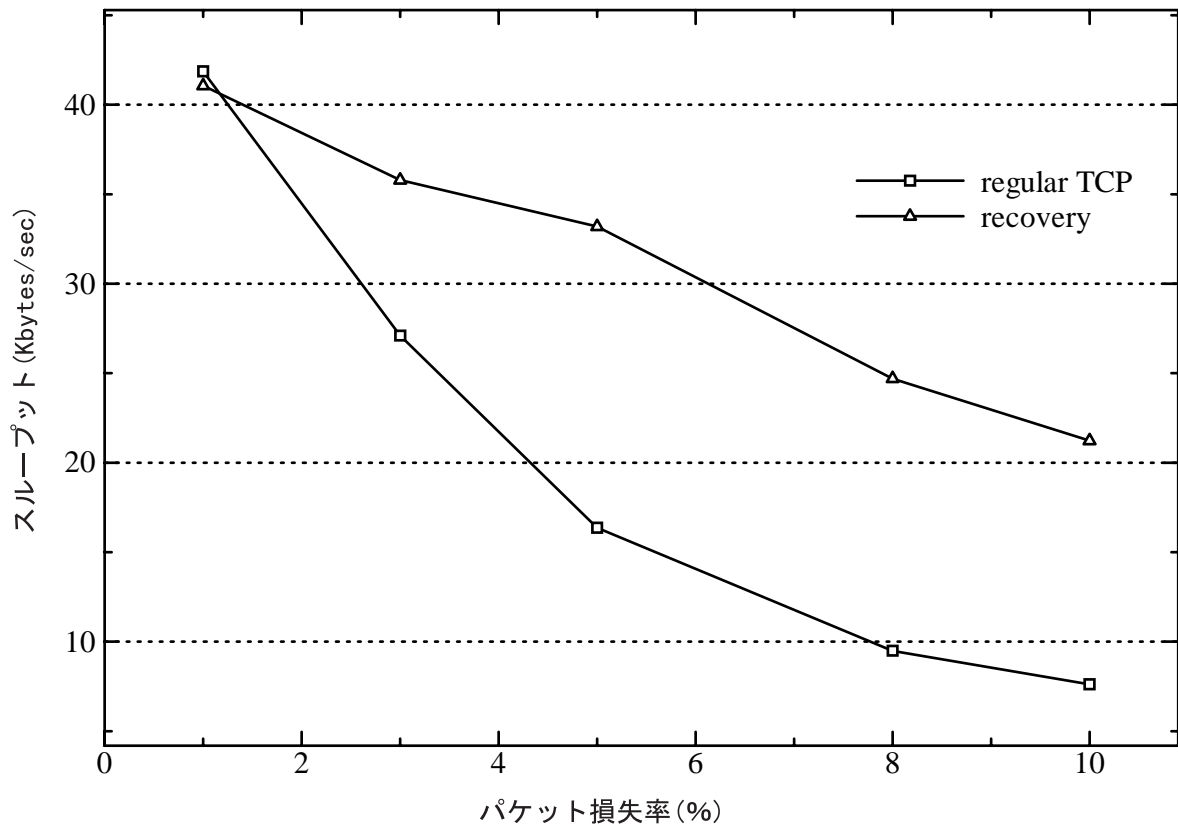


図 6.4: エラー率に対するスループットの推移

図 6.2 は、パケットの損失がほとんど発生しない状況において、4K バイトから 1024K バイトまでの各サイズのデータを転送した場合のスループットの推移である。32K バイト以下のデータ転送では、TCP のスロースタートと輻輳回避の影響を残しているが、データサイズが 64K バイトを越えると、それらの影響は隠され、定常状態のスループットに達している。また、パケットの損失がほとんど発生しない状況では、通常の TCP とリカバリモジュールを使用した場合に、ほとんど差は現れない。

パケットの損失率を 5% に設定し、同様の測定を行った結果が図 6.3 である。さらに、パケット損失率 3%、8%、10% に対して同様の測定を行い、これらの場合においても、データサイズが 64K バイト以上になれば、スループットが安定することを確認した。この結果から得られたパケット損失率に対するスループットの推移を図 6.4 に示す。グラフ上の各点は、各パケット損失率において、64K バイト、128K バイト、256K バイト、1024K バイトのデータ転送を行った結果の平均スループットをプロットしたものである。グラフから、通常の TCP はパケット損失率の上昇に対して、スループットを極端に下げってしまう

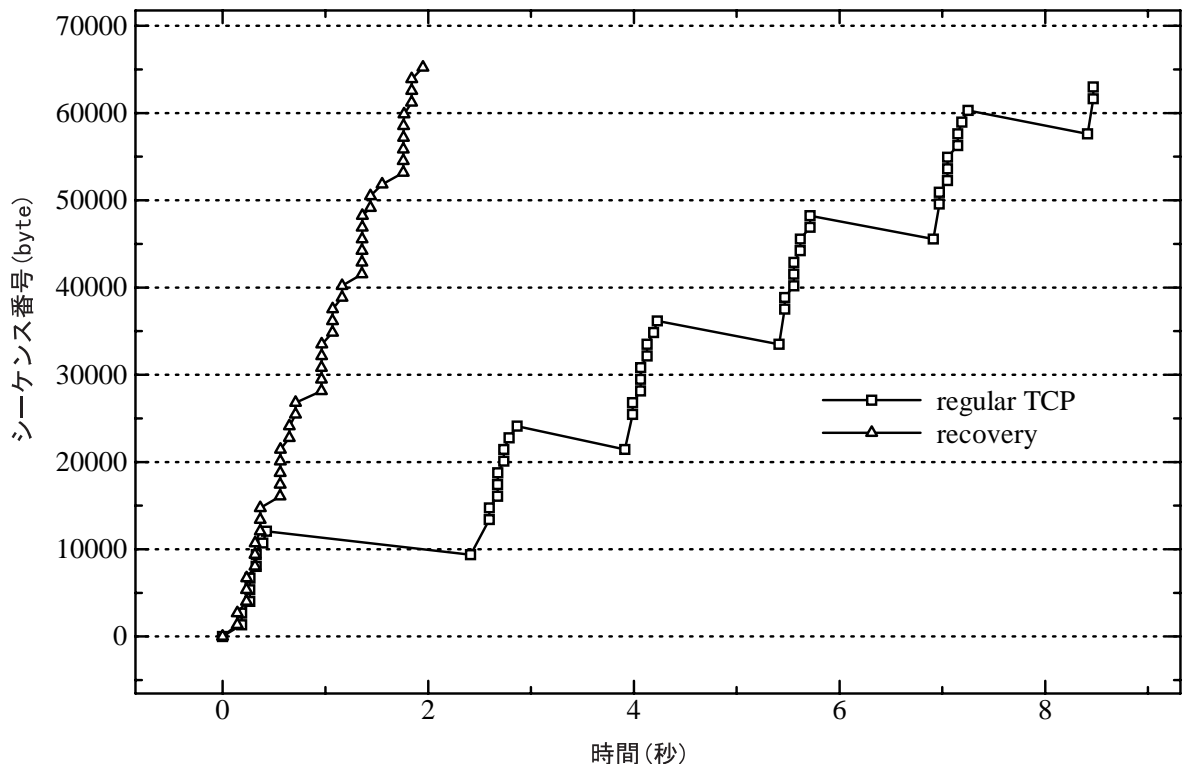


図 6.5: 64K バイトのデータ転送におけるシーケンス番号の伸び

のに対し、リカバリモジュールを用いた場合は、スループットの低下を十分に抑制できることが分かる。

この差が具体的にどのような状況で生じるのかを検証するために、64K バイトのデータ転送の詳細な様子を観測した。結果を図 6.5 に示す。この測定では、送信元から 10 個の packets が送られるごとに、その 10 番目の packet を人為的に落としている。グラフから、通常の TCP では、送信側のウィンドウが十分に開かないうちに packet が損失すると、極端に性能が落ちてしまうことがわかる。TCP に備えられているリカバリメカニズムにおいても、重複 ack に基づいた高速な再転送を実現しているが、エンド - エンドのリカバリメカニズムであるため、packet の損失を正確に検出することを難しくしている。

送信側は、1 つ目の重複 ack を受け取った段階では、それが、packet の順番が入れ替わって届いたことが原因で返された重複 ack なのか、実際に packet が失われたことによって返された重複 ack なのかを判定できない。2 つ目の重複 ack が届いた時点でようやく、packet が損失したものと判断している。しかし、ウィンドウが開ききらない状態では、この 2 つ目の重複 ack が返されない状況が発生する (送信側のウィンドウサイズの制

限で、失われたパケットに続いてパケットを送信できていない場合)。このとき、送信側の TCP は、再転送のタイマが切れるのを待つ以外にない。図 6.5において、通常の TCP がパケットの再転送を行うまでに、極端な遅延を生じているのは、これが原因であると考えられる。

一方、パケットスヌーピングによるリカバリメカニズムでは、プロキシエージェントにおいて送信元からのパケットをスヌーピングする時点で、順番が入れ替わっているパケットを記録している。このため、一つ目の重複 ack をプロキシエージェントで検出し、それに対応する TCP セグメントをキャッシュ中に発見すると、その時点で再転送を行うことが可能であり、パケットの損失によるスループットの低下を最小限に抑えることができる。

6.2 考察

前節の評価結果は、固定された状況におけるモジュール単体の性能評価である。パケットスヌーピング機構の本来の評価としては、これらのモジュールを状況に応じて切替え、通信メディアの特性にいかにか柔軟に適應することができるかを示す必要がある。

モジュールの切替および切り離しに関しては、前章で述べたスヌーププロトコルによって、移動計算機からの要求を即座に反映させることができるため、極端な例としては、パケット毎に最適化モジュールを切替えることも可能である。パケットスヌーピング機構のこういった特徴は、無線 LAN のように、状況に応じて特性が変化するメディアに対して非常に有効である。パケットの損失が発生しない安定した状態において、無線 LAN は比較的狭いバンド幅を持つ通信メディアに分類される。このとき、パケットを十分に圧縮できる場合には、圧縮モジュールの使用が効果的である。一方、パケットの損失率が高くなる状況では、エラーリカバリモジュールを使用する。こういった方針を採用する場合に、パケットの圧縮率や損失率の変化をどのように把握するかが問題となる。圧縮率に関しては、移動計算機の圧縮モジュールにパケットの圧縮率をサンプリングする機能を持たせることで対処し、パケットの損失率に関しては、環境サーバから提供される情報を利用することができる。また、パケット圧縮処理に要するオーバーヘッドは、移動計算機とプロキシエージェントで異なるため、これを反映させるには、移動計算機とプロキシエージェントの圧縮モジュールがプロセッサの処理速度などの情報を交換する必要がある。この種の情報の交換は、スヌープヘッダに続いて圧縮モジュール専用のヘッダを用意することで対処することができる。

以上は今後の課題となるが、現状においても、物理的に通信メディアを切替えた場合の特性変化への適應は可能である。たとえば、図 6.1の環境で、移動計算機のネットワーク

カードをイーサネットカードに差し替えると、移動計算機内のメディアセクタがそれを判断し、メディアの切替えと同時に最適化モジュールおよびプロキシエージェントを即座に切り離す。以降は、移動計算機内のフォーリンエージェントエミュレータがホームエージェントからのカプセル化パケットを直接受け取る。このように、パケットスヌーピング機構は変化の激しい移動計算機環境に適応し、効率の良いデータ転送を実現している。

また、本機構は、パフォーマンスの追求に関して、従来のシステムとはまったくことなれたアプローチを取っている。たとえば、第2章で関連研究としてあげたバークレースヌーププロトコルは、カーネルのパケットバッファに変更を加えることで、エラー発生率が低い状況でのエラーリカバリのオーバーヘッドを避けようとしている。一方、本機構は、エラーリカバリモジュールを使用している状況で、エラー発生率が無視できるほど小さくなったとメディアセクタが判断すると、パケットスヌーピング機構はその判断に連動し、エラーリカバリモジュールを切り離す。したがって、不必要なエラーリカバリ処理をまったく排除することができる。つまり、様々な最適化ポリシーをモジュール化し、それらを動的に切替えるという拡張性の高い設計が、結果的にパフォーマンスの向上に結びついていると言える。

第 7 章

おわりに

移動計算機環境を構築するための従来のアーキテクチャは近年の通信メディアの多様化に十分に対応していない。

本論文では、多様化した通信メディアに見合う移動計算機環境の在り方を示し、それを実現するためには、従来のアーキテクチャでは根本的な問題点が残されていることを指摘した。また、それらの問題点を解決するための方法を示し、改善点を全て備えた統合的なアーキテクチャとして、JAIST Mobile IP システムを導入した。JAIST Mobile IP システムは、通信メディアの多様化に対応した移動計算機環境を実現する上で主要となる以下の 4 つの機能を実現する。

- コネクションを維持した状態での計算機の円滑な移動
- 適切な通信メディアへの切り替え
- 通信メディア切り替えのタイミングの制御
- 通信メディア特性の変化への適応

本論文では、とくに「通信メディア特性の変化への適応」に焦点を当て、メディア切替や計算機の移動によって発生するメディアの特性、すなわちエラー率、バンド幅、遅延などの変化への適応を可能にする機構の設計と実装について議論した。本機構は、移動計算機宛てのパケットのスヌーピングをベースとしており、通信メディアの特性に合わせて作成された最適化モジュールをメディアの特性の変化に応じて切り替えることで、移動計算機環境における TCP/IP によるデータ転送の効率を最大限に改善する。また、最適化モジュールのインターフェースを統一することで、新しいモジュールの追加と変更、すなわち最適化ポリシーの追加と変更を容易にしている。

また、本機構を組み込んだ JAIST Mobile IP システムを実際に稼働させるために、Real-Time Mach 上への実装を行った。システムの評価としては、最適化モジュールのテストケースとしてパケット圧縮モジュール、エラーリカバリモジュールを作成し、その動作と効果を示した。測定の結果、それぞれのモジュールは十分に機能し、本機構が、通信メディアの特性を考慮したモバイルネットワークを実現する上で有効であることを実証した。また、その際に、拡張可能システムの有用性について議論し、JAIST Mobile IP システムが、可能な限り拡張性を維持しながら、十分なパフォーマンスの向上を実現するシステムであることも示した。

今後の課題としては、様々な特性に応じた最適化モジュールの開発、アプリケーションに依存した最適化、モジュールを動的に挿入する機構の追加といった項目が考えられる。最適化モジュールの開発には、こういった特性に対してこういったモジュールが有効であるのかを見極めることが重要になる。たとえば、往路と復路でバンド幅や遅延の異なる通信メディアにどのように対応するかといった問題が考えられる。データ転送の最適化以外にも、たとえば無線を使用しているときには、パケットの内容を暗号化するモジュールの必要性も考えられる。

また、アプリケーションに依存した最適化を可能にすることを考える場合、パケットのスヌーピングに基づいた方法では、アプリケーションが転送するデータの形式を特定できることが前提となる。しかし、個々のアプリケーションのデータ形式毎に最適化モジュールを用意することは現実的ではない。アプリケーションを起動する移動計算機にそれらのモジュールを組み込むことに抵抗はないが、プロキシエージェントに各移動計算機が使用するアプリケーションを予測した上で、様々な最適化モジュールを用意しておくことは非効率的である。そこで、プロキシエージェントが持つベーシックな最適化モジュール以外に、移動計算機から送られてきたモジュールを動的に挿入し、切替える機能をプロキシエージェントのパケットスヌーピング機構に追加する。JAVA で用いられている技法などを利用することで、状況に応じて振る舞いを変えるソフトウェアを作成することは可能である。

今後は、以上のような拡張を取り入れながら、より柔軟な移動計算機環境の実現を目指して、JAIST Mobile IP システムを発展させていきたい。

参考文献

- [1] C. Perkins, “IP Mobility Support”, RFC2002, October, 1996.
- [2] D.A.Maltz, D.B.Johnson, “The CMU Monarch Project IETF Mobile IPv4 Implementation User’s Guide”, <http://www.monarch.cs.cmu.edu/>, February, 1997.
- [3] R.Drome, “Dynamic Host Configuration Protocol”, RFC1541, October, 1993.
- [4] S.E.Deering, “ICMP router discovery messages”, RFC1256, September, 1991.
- [5] Perkins, C., “IP Encapsulation within IP”, RFC 2003, October, 1996.
- [6] 中島達夫, “モバイルコンピューティングのための動的適応可能なソフトウェアアーキテクチャ”, 第 16 回 IPA 技術発表会, 1997. James D. Solomon,
- [7] Hari Balakrishnan, Srinivasan Seshan, Randy H.Katz, “Improving Reliable Transport and Handoff performance in Cellular Wireless Networks”, ACM Wireless Networks, December 1995.
- [8] 小林勝, 中島達夫, “動的なメディア選択が可能な Mobile IP の設計と実装”, IPSJ OS, February, 1998.
- [9] Tatu Nakajima, Hiroyuki Aizu, Masaru Kobayashi, Kenji Shimamoto, “Environment Server: A System Support for Adaptive Distributed Application”, The 2nd International Conference on Worldwide Computing and Its Applications’98 (WWCA’98), March, 1998.
- [10] 森川大樹, 植田道成, 石橋賢二, 中島達夫, “通信メディアの特性を利用する Mobile IP システム”, Summer United Workshops on Parallel, Distributed, and Cooperative Processing (SWoPP), August, 1998.

- [11] T. Tokuda, T. Nakajima, and P. Rao, “Real-Time Mach: Towards a Predictable Real-Time System”, Proceedings of the 1st USENIX Mach Symposium, October, 1990.
- [12] J. Helander, “Unix under Mach: The Lites Server”, Helsinki University of Technology, Master’s Thesis, 1994.