マルチメディア通信と分散処理 102−22
（2001．3．22）

複数無線リンク上でのコネクション型通信手法の性能評価

飯田 峰彦† 石原 進‡ 水野 忠則‡
静岡大学大学院理工学研究科† 静岡大学情報学部‡

筆者らは無線通信環境で論理的に帯域を広くし，より信頼性の高い通信を実現するための方式として通信回線共有方式を提案している．通信回線共有方式は携帯端末を複数集め，これらを相互接続してネットワーク（クラスタ）を構築する．各端末はクラスタのための回線とISP, LANなどの外部ネットワークへの接続回線を持つ．クラスタを構成する端末が外部ネットワークへ一つのデータを送受信する時に複数の外部通信経路を同時に利用，かつ共有することで，無線通信環境での通信速度の向上を図る方式である．筆者らは，本方式をアプリケーション層で複数TCPコネクションを確立し，通信を行う試作システムSHAKEを開発している．また，TCP層で複数経路の存在を考慮した信頼性保証のための制御を行うコネクション型通信手法 *TCP SHAKE* を提案している．本稿ではTCP SHAKEの性能評価を行った．その結果，TCPで各経路に割り当てた輻輳ウィンドウに基づいて振り分けを制御することで，各経路の品質変動に柔軟に対応できることが示された．

Performance Evaluation of the method of connection-type communication on multiple wireless links

Minehiko Iida† , Susumu Ishihara‡ and Tadanori Mizuno‡

We have proposed "Sharing Multiple Paths System". In Sharing Multiple Paths System, some mobile hosts are connected each other and form a local network, which we call *cluster*. Each mobile host has a channel connected with other mobile hosts and a channel connected with outside network such as Internet or LAN. When a mobile host sends data to a destination in the outside network, data transmission speed becomes higher even in the wireless communication environments by using multiple paths at the same time and sharing it with some mobile hosts. We have developed the system "SHAKE", which communicates by establishing multiple TCP connections. We also propose the effective connection-type communication strategy on multiple wireless links *TCP SHAKE*. In this paper, we evaluate the performance of TCP SHAKE. As a result of evaluation by simulation, we found it can deal with the condition changes of multiple links flexibly.

1 はじめに

今日のノートPC, PDAといった情報携帯端末の小型化，高性能化および，携帯電話, PHSといった無線通信インフラの爆発的な普及と通信速度の向上により，出先や移動中であっても気軽に，誰とでもマルチメディアコンテンツを交換することが一般的になってきた．また，携帯電話のユーザに提供されるサービスの多様化によってモバイルコンピューティングを取り巻く環境も充実しつつある．しかしながら，無線通信は速度が向上したとは言え，有線通信に比べると依然として帯域が狭く，品質が不安定であるという特性があるため，大量のファイル転送やリアルタイム性を要求されるようなマルチメディアデータの通信は困難である．

そこで，筆者らは図1に示すように携帯電話などの無線通信回線を持つ携帯情報端末が集まり，それぞれが持つ無線通信帯域を共有することで各端末が広い帯域を使用可能にし，より信頼性のある通信を実現する方式として通信回線共有方式を提案している [1]．

複数経路を並列利用することで転送速度を向上させるための研究として，通信を行う両ホストの間に複数経路にパケットを振り分ける機能を持った専用のゲートウェイを設置する複数経路データ転送方式 [2]，データリンク層での並列配信を実現する384KbpsのPHS接続装置を用いた通信方式 [3] および，PPPマルチリンク [4] などがある．これらに対し，本稿で論じる通信回線共有方式はトランスポート層で複数経路をサポートする [5]．

筆者らは，通信回線共有方式SHAKE (*Shared mulitilinK procedures for a cluster type networe Environment*) の試作システムを開発している．この実装では複数の経路を複数のTCPコネクションを取りまとめるライブラリで制御した．試作システムによる実験で，複数経路を利用することによる転送速度の向上を確認できたが，TCPコネクションを複数確立することによるオーバヘッドのために，経路数を n にすることによって n 倍の転送速度は得られなかった．この実装では，再送経路の選択に自由度がなく，各TCPコネクションへの動的な振り分け率の変更に制限があるなどの問題があった．そこで，筆者らはこの問題を

† Graduate School of Science and Engineering, Shizuoka University
‡ Faculty of Information, Shizuoka University

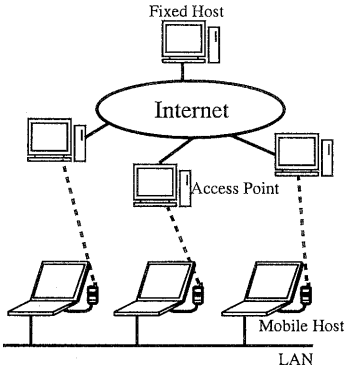

図 1 通信回線共有方式の構成例

解決するためのアプローチとして，TCP で複数経路を考慮した上で信頼性を保証するための制御を行うコネクション型通信手法 *TCP SHAKE* を提案している [5]．本稿では，TCP SHAKE の性能評価を行う．

2 TCP SHAKE

2.1 基本方針

TCP SHAKE は，クラスタメンバが通信相手と単一のコネクションを確立し，トランスポート層で複数経路にセグメントを送受信する End-to-End の通信環境を提供する．

TCP SHAKE は，送受信バッファを 1 つずつ持ち，TCP の拡張として信頼性を保証するための再送制御と輻輳制御を経路毎で行うために送信側の各経路で輻輳ウィンドウ，再送タイマを用意する（図 **2**，図 **3**）．セグメントを送信するとき，一度に送信できるデータは各経路の輻輳ウィンドウによって制御される．また，受信側では受信したセグメントと同じ経路で ACK を送信する肯定確認応答を利用する．

2.2 輻輳制御

2.2.1 経路毎の輻輳ウィンドウ

TCP SHAKE では輻輳制御において，各経路に割り当てられた輻輳ウィンドウに従い，一度に送信可能なデータサイズを制御する．TCP SHAKE の各経路の輻輳ウィンドウの和が従来の TCP の輻輳ウィンドウに相当する（図 **2**）．この値と受信側が受信可能なデータサイズを表す受信ウィンドウ（図 **3**）を利用し，その小さい方が送信側の各経路で一度に送信できるデータサイズとして利用される．

送信ウィンドウは三つのパラメータで管理される．*snd_una* は ACK を受信していないセグメントの先頭

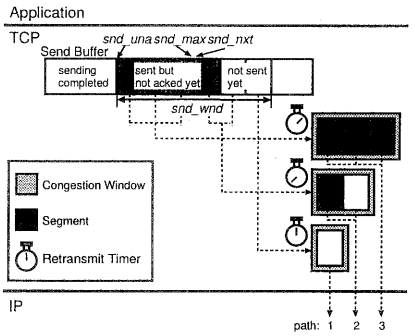

図 2 TCP SHAKE の概念図（送信側）

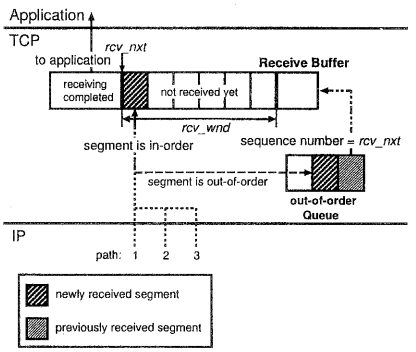

図 3 TCP SHAKE の概念図（受信側）

番号を表す．新しい ACK を受信することでこの値は増やされる．*snd_max* は送信済みセグメントの末尾番号を表す．*snd_nxt* は次に送信するセグメントの先頭番号を表す．この値が TCP ヘッダ内のシーケンス番号の値になる．また，再送タイムアウトが発生すると *snd_una* と同じ値に戻される．一方，受信側では次節で述べる SACK の利用を除いて従来の TCP と同様である．

各経路の輻輳ウィンドウはコネクション確立時に初期値 (MSS) に設定され，送信側が ACK を受信する度に輻輳ウィンドウを増加する．タイムアウトが発生すると輻輳ウィンドウは初期値 (MSS) に戻される．なお，各経路の輻輳ウィンドウの制御は 4.3BSD 以降の TCP Reno のスロースタートと輻輳回避アルゴリズムに従う．

2.2.2 輻輳ウィンドウに従った振り分け

TCP SHAKE ではセグメントはそれぞれ異なる特性を持った経路を通るため，送信時にシーケンス番号の若いセグメントから順に送信しても，受信側で順序どおりに連続したセグメントを受信できる保証はない．セグメントが非連続で受信されると重複 ACK

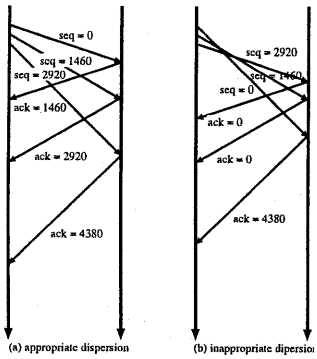

図 4 振り分け方式の違い

が頻繁に発生し，その結果無駄な再送を引き起こし，スループットは著しく低下する（図4(b)）．そこで受信側で順序どおりにセグメントが受信できるように，送信側は各経路に割り当てられた輻輳ウィンドウサイズにしたがって，帯域が広いとみなされる経路からシーケンス番号の若いセグメントを順に送信する（図4(a)）．したがって，振り分け率は各経路の輻輳ウィンドウのサイズの比に従い，時間によって変化する輻輳状況に応じて動的に変化する．

2.3 再送制御

再送タイムアウトの時間内にACKが受信されないと，セグメントは再送される．各経路でRTTを測定し，それをもとに各経路で再送タイムアウトを算出する．なお，算出アルゴリズムは4.3BSD以降のTCP Renoの実装に従う．送信側が経路毎のRTTを測定するために，受信側はセグメント毎に，受信したセグメントが通ってきた経路と同じ経路でACKを送信する．したがってDelayed ACKは使用しない．送信側では送信したセグメント数と受信したACKの数を経路毎に管理しており，再送タイムアウト以内に送信したセグメントがすべて確認通知されない経路では再送が発生する．セグメントを再送する時，再送セグメントは前節の振り分け方式に従い，帯域が最大の経路で送信する．再送するセグメント数は帯域が最大の経路のウィンドウ幅に従う．

TCP SHAKEでは各セグメントは異なる経路を通って受信側に送られるため，セグメントが不連続に受信される可能性が単一経路での通信よりも高くなる．受信側がセグメントを断続的に受信すると，セグメント送信側は重複ACKを頻繁に受信することになる．その後，タイムアウトが発生すると snd_nxt は snd_una と同じ値に戻される．このとき送信側は

$$snd\_nxt < seqnum < snd\_max$$

のシーケンス番号を持つセグメントを一度送信している．しかしながら，それらのセグメントのうち，どのセグメントが受信され，どのセグメントがロスしたのかを送信側は詳細に知ることができない．したがって，再送時にすでに受信側が受信しているはずのセグメントを複数送信する可能性があり，転送が重複する．そこで，TCP SHAKEではSACK(*Selective Acknowledgement*)[6]を利用する．SACKはセグメントの受信側が断片的に受信したセグメントブロックの情報をACKのTCPオプションフィールドに付加することで，送信側により詳細な受信状況を知らせる．送信側は，輻輳ウィンドウと受信側の受信状況から送信セグメントの数を決定する．

3 性能評価

3.1 シミュレーション

TCP SHAKEの無線通信環境での転送レート，対障害性を検証するためにシミュレーションを行った．シミュレーションでは，表1に示すように経路1が最も帯域が広く，他の2経路は経路1の半分の帯域を持つような3つの経路で通信を行うことを想定している．また再送セグメントを，遅延が短く，かつエラー率が低い経路で送ることによる効果を検証するために経路1の通信品質は全ての評価を通じて理想的な品質状態に設定してある．シミュレーションモデルを図5に示す．[5]では再送経路の選択，およびSACK使用による効果について評価した．本稿では，さらに

- パケットロス率，バーストエラーによる影響
- 輻輳ウィンドウの推移
- マルチTCPコネクションによる通信との比較

に関して評価を行った．これらの結果を以降で示す．

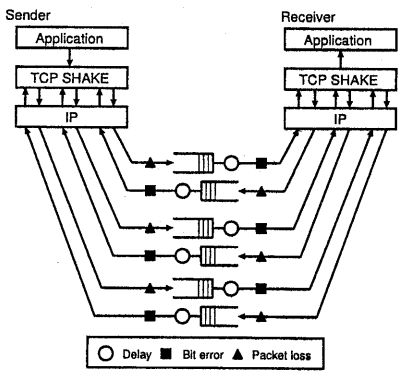

図 5 TCP SHAKE シミュレーションモデル

表 1 3つの経路の帯域と平均遅延

	帯域 (Kbps)	平均遅延 (sec)
経路 1	64	0.13
経路 2	32	0.1875
経路 3	32	0.1875

3.2 結果と考察

3.2.1 再送経路選択，SACK の評価

各経路のパケットロス，ビットエラーが表 **2** の構成で 1000(Kbyte) のデータを転送したときの，

- 再送経路の選択による効果
- SACK の効果

を評価した．なお，前者の評価では SACK を使用した上で再送セグメントの送信経路の選択手法の違いによる比較を，後者は再送経路として帯域が最大の経路を利用した上で SACK 使用時と不使用時の比較を行った．MSS のサイズは 1460 (byte) である．それぞれの結果を図 **6** と図 **7** に示す．二つのグラフは図 **3** における *rcv_nxt* の時間変化を表している．

表 2 シミュレーション条件

	packet loss (%)	bit error(%)
経路 1	1	1
経路 2	4	1
経路 3	4	1

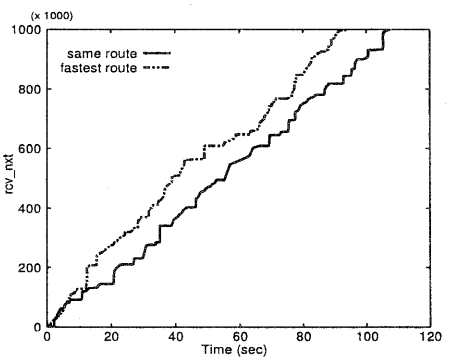

図 6 再送経路選択の効果

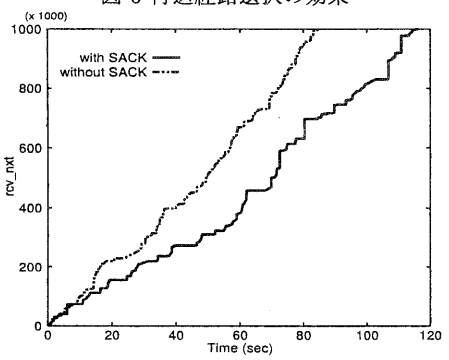

図 7 SACK 使用の効果

再送経路の選択に関しては，任意の経路で再送タイムアウトによって再送が発生したとき，再送セグメントを送信する経路を帯域が広い経路に切り替えることによる性能向上が見られた．今回は再送セグメントを送信する経路の判断基準として，コネクション確立時にクラスタメンバに関する情報から得られる帯域幅の情報のみを利用した．しかしながら，帯域は広いがパケットロス率が高いような通信環境ではこの再送方式は適用しにくい．再送や振り分けの判断基準となる指標は帯域幅だけではなく，パケットロス率も考慮する必要がある．帯域以外の指標としては *cwnd* か RTT が考えられる．

SACK 不使用時には *rcv_nxt* の値が停滞する時間が長く，不連続もしくは重複受信が頻繁に発生している一方で，SACK を使用することで再送時の無駄な重複セグメントの受信が起こる割合が低くなり，*rcv_nxt* はほとんど停滞することなく，ほぼ一定のスループットでセグメントを受信している．

3.2.2 パケットロス，バーストエラーによる影響

パケットロスによる影響

表 **3** のように，各経路のパケットロス，ビットエラー率の組み合わせが異なる 3 つのシナリオにおいて 100(Kbyte) から 1000(Kbyte) までのデータを送信したときのデータ転送時間を測定した．TCP SHAKE は SACK を使用し，再送セグメントは帯域が最大の経路で送信している．測定結果を図 **8** に示す．なお，測定は各シナリオにおける転送データサイズに対して 50 回測定した平均値を示している．

各シナリオ間で，データ転送時間に差はほとんど見られなかった．TCP SHAKE での制御によって，再送時に適切な経路を選択した上で再送セグメントを送信し，かつ SACK を利用することで無駄なトラヒックを最小限に抑えられていることと，輻輳ウィンドウを用いて各経路でフロー制御が適切に行われている

表 3 シミュレーション条件

シナリオ **1**	packet loss (%)	bit error (%)
経路 1	4	1
経路 2	4	1
経路 3	4	1
シナリオ 2	packet loss (%)	bit error (%)
経路 1	1	1
経路 2	4	1
経路 3	4	1
シナリオ 3	packet loss (%)	bit error (%)
経路 1	1	1
経路 2	4	1
経路 3	2	1

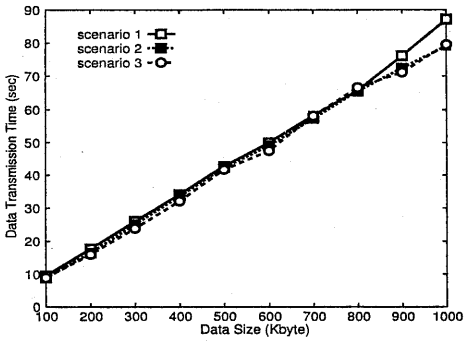

図 8 パケットロス率の違いによる比較

表 4 シミュレーション条件

送信データサイズ (byte)	500,000
経路 1 の平均バースト長 (segment)	1
各経路の平均バースト間隔 (segment)	50

表 5 バーストエラー評価におけるシナリオ

	再送経路	SACK
シナリオ 1	帯域が最大の経路	不使用
シナリオ 2	帯域が最大の経路	使用
シナリオ 3	同じ経路	不使用
シナリオ 4	同じ経路	使用

ことから各経路のパケットロス率の組合せが変化しても，一定のスループットを得られると考えられる．

バーストエラーによる影響

次に表 4 の条件下で経路 2, 3 の平均バースト長を 1〜5 まで変化させたときのそれぞれの送信時間を測定した．また，この測定を表 5 に示す 4 つのシナリオで比較した．結果を図 9 に示す．なお，グラフは各平均バースト長に対して 100 回の測定を行った平均値を示している．

図 9 から，シナリオ 3, 4 の方がシナリオ 1, 2 よりも短い時間で送信できている．したがって，再送経路を適切に選択することによる効果が現れていることがわかる．一方，SACK による効果は顕著に見られなかった．バーストエラーによってセグメントがいくつか連続してロスされると，再送後転送が重複するセグメント数が少ない．すなわちセグメントの重複転送が原因で転送速度が低下するわけではないため，SACK を利用することによる効果が現れにくいと考えられる．

3.2.3 輻輳ウィンドウの推移

各経路のパケットロス率が表 6 に従い，1,000 (kbyte) のデータを送信したときの各経路の輻輳ウィンドウの時間変化を測定した．このシミュレーションでは，TCP

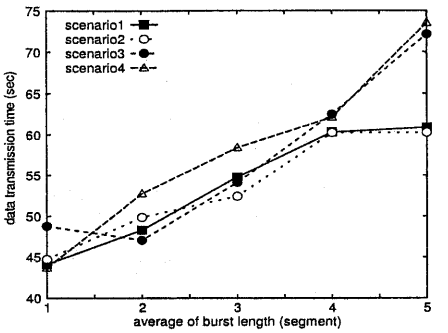

図 9 バーストエラーの評価

表 6 シミュレーション条件

	packet loss (%)	bit error (%)
経路 1	1	1
経路 2	4	1
経路 3	4	1

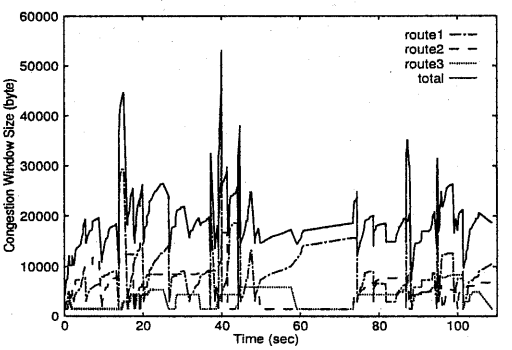

図 10 輻輳ウィンドウの推移

SHAKE は SACK を利用し，再送セグメントは帯域が最大の経路で送信している．結果を図 10 に示す．

経路 1 はパケットロス率が最も低く，かつ帯域が最も広いため，セグメントが多く送信される．このため輻輳ウィンドウが最も大きく，変動も激しい．一方，経路 2 と経路 3 は帯域が経路 1 よりも狭く，かつパケットロス率も高いため，輻輳ウィンドウが大きくならず，低い値を維持している．各経路での輻輳ウィンドウは従来のスロースタート，輻輳回避アルゴリズムを採用しているため，再送タイムアウトが発生すると輻輳ウィンドウは最小値 (MSS) にまで下がってしまう．しかしながら，3 つの経路全体として見ると変動が激しいが任意の経路で再送タイムアウトが発生したときに輻輳ウィンドウが最小値にまで下がることはなく，経路全体としては帯域を有効に利用していると考えられる．

表 7 シミュレーション条件

	ロス率 (%)
経路 1	3
経路 2	5
経路 3	4

3.2.4 マルチコネクションを利用した通信との比較

TCP SHAKE と SHAKE プロトタイプのような複数の TCP コネクションを同時に複数確立して通信した場合の比較を行った．**表 7** に示す条件において

- マルチコネクションで帯域幅の比に応じたトラヒック分配を行った場合
- マルチコネクションで等分配して振り分けた場合
- TCP SHAKE で通信

の 3 つの場合で 100(Kbyte) から 1000(Kbyte) のデータを送信したときの送信時間を 50 回測定したときの平均値を比較した．シミュレーションにおける TCP のパラメータを**表 8** に示す．また，それぞれの測定結果を図 **11** に示す．送受信バッファサイズの値は 3 本の複数コネクションでの各値の和が TCP SHAKE 使用時とほぼ等価になるように設定している．なお，グラフ内の multi connection1 はマルチコネクション環境でパケットを等分配した場合（振り分け比 1:1:1），multi connection2 はパケットを帯域幅を重みとして振り分けた場合（振り分け比 2:1:1）である．

マルチ TCP コネクション環境で帯域幅を重みとして振り分けた場合（multi connection2）と TCP SHAKE ではほとんど差が見られなかった．アプリケーションで帯域幅を意識して振り分けを行っているため，TCP SHAKE との差が小さくなったと予想される．これに対し，マルチ TCP コネクション環境でアプリケーションが帯域幅を意識せずにパケットを各コネクションに等分配するとスループットが低下した．これは帯域が広い経路と狭い経路に同じサイズのデータを送っているため，複数回線を同時に利用してもスループットは帯域が狭い経路に依存してしまうためである．

マルチコネクションの通信環境であっても，アプリケーションが帯域幅を考慮した振り分けを行えば，比較的高いスループットが得られることがわかった．しかしながら，アプリケーションが常に各経路の帯域を知ることができるとは限らないため，常に適切な振り分けができるとは言えない．一方，TCP SHAKE では TCP で輻輳ウィンドウによって，遅延の変動に応じて各経路のフロー制御を行っているため，無線通信環境であっても常に最適な振り分けが可能になる．例えば，各経路の帯域が通信中に変化するような状況で有効である．

表 8 TCP に関わるパラメータ

パラメータ	値	
	TCP SHAKE	複数コネクション
send buffer(byte)	52560	17520
receive buffer(byte)	52560	17520
Delayed ACK	off	on
tcprexmtthresh	10	3
snd_ssthresh(byte)	29200	29200

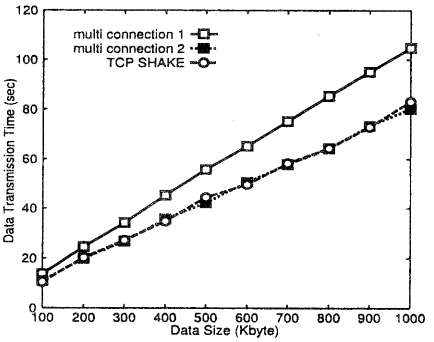

図 11 3 つの通信方式におけるデータ送信時間

4 おわりに

本稿では，TCP SHAKE の性能評価を行った．その結果，輻輳制御，再送制御において複数経路を効率的に利用することでセグメントの無駄な送信が減り，遅延の変動に対応した適切な振り分けが可能になることを示した．また，SACK の利用，再送経路の適切な選択により，パケットロスやバーストエラーにも対応できることを示した．

今後は，再送時に再送セグメントを送信する経路を決定するパラメータを帯域のみではなく，パケットロス，RTT などの通信特性を示すパラメータも考慮した方式を検討する予定である．

参考文献

[1] H.Mineno, S.Ishihara, K.Ohta, M.Aono and T.Mizuno: Multiple paths protocol for a cluster type network, International journal of communication systems (1999).

[2] 星谷直哉, 相田 仁, 斎藤忠夫: 複数経路データ転送におけるコネクション指向プロトコルの提案, 第 59 回情処全大, 1T-7, pp.349-350 (1999).

[3] 神尾享秀, 児島史秀, 藤瀬雅行: 384kbps-PHS 実験装置の概要と性能評価, 情処研報 Vol.99, No.80, 99-MBL-10 (1999).

[4] K.Sklower, B.Lolyd, G.McGregor and D.Carr: The PPP Multilink Protocol (MP), RFC 1717 (1994).

[5] 飯田峰彦, 石原 進, 水野忠則: 複数無線リンク上での効率的コネクション型通信手法の提案: 情処研報 Vol.2000, No.87, 2000-MBL-14 (2000).

[6] M.Mathis, J.Mahdavi, S.Floyd and A.Romanow: TCP Selective Acknowledgment Options, RFC 2018 (1996).