

無線ネットワーク環境に適したトランスポートプロトコルTCP-Jの 実装, 解析および評価

斉藤 俊介[†] 寺岡 文男[†]

[†] 〒223-0061 神奈川県横浜市港北区日吉 3-14-1
慶應義塾大学大学院理工学研究科 寺岡研究室

E-mail: [†]shun@tera.ics.keio.ac.jp, ^{††}tera@ics.keio.ac.jp

あらまし 本稿では、無線ネットワーク環境に適したトランスポートプロトコル TCP-J を NetBSD1.6.1-Release のカーネル内に実装し、dummynet により無線区間をエミュレートすることで既存の TCP との性能比較を行った。その結果、TCP-J は無線リンクでのランダムなセグメント損失率が 2×10^{-2} のとき、NewReno と比べて約 128 %, SACK と比べて約 110 % のスループット向上を得ることができた。また、TCP-J はセグメント損失率が 0 の場合は従来 TCP と公平にバンド幅を分け合い、セグメント損失率が増加して従来 TCP のスループット低下によって無線リンク上のバンド幅に空きができた場合には、他の TCP 通信を圧迫せず、その帯域を有効利用してスループットを改善できることが分かった。

キーワード TCP, 輻輳制御, 無線ネットワーク環境

Implementation, Analysis and Evaluation of TCP-J: A new version of TCP for wireless networks

Shunsuke SAITO[†] and Fumio TERAOKA[†]

[†] 3-14-1 Hiyoshi, Kohoku-ku, Kanagawa 223-0061 Japan
Teraoka Laboratory, Graduate School of Science and Technology, Keio University

E-mail: [†]shun@tera.ics.keio.ac.jp, ^{††}tera@ics.keio.ac.jp

Abstract In this research we implemented a new version of TCP - TCP-J, improving TCP performance over wireless networks and evaluated TCP-J performance against existing versions of the TCP. In result, when the segment loss rate was 2×10^{-2} , TCP-J increased throughput by 162 % compared with Reno, by 128 % compared with NewReno, by 110 % compared with SACK. TCP-J fairly shared the bandwidth with the conventional TCP (NewReno, SACK) when the segment loss rate was 0.

Key words TCP, Congestion Control, Wireless Network Environment

1. はじめに

近年、無線 LAN や携帯電話によるインターネット利用により、無線を介したインターネットアクセスが急速に増加してきている。また、現在のインターネットトラフィックの大部分は Web ブラウジングと電子メールの送受信であり、これらのアプリケーションはトランスポートプロトコルに TCP (Transmission Control Protocol) を利用しているため、今後は無線を介した TCP 通信が増加していくと思われる。

有線ネットワークでは通信回線が非常に安定しているため、TCP セグメントの損失がネットワークの混雑 (輻輳) 以外の原因で発生する可能性は極めて低い。そのため、既存の TCP はセグメントの損失を輻輳によるものと判断し、輻輳ウィンドウを狭めることによってデータの転送量を抑え、輻輳状態を解消する。一方、無線ネットワークでは、雑音や電波干渉などによる高いビット誤り率に起因したセグメント損失や、端末の基地局間の移動に伴うセグメント損失など、有線ネットワークとは異なり、輻輳以外のことでセグメント損失が多数生じる。このような無線ネットワーク特有のセグメント損失は輻輳とは無関係に生じるため、輻輳制御を行う必要がない。しかし、従来の TCP はセグメントの損失を一律に輻輳と判断して大きくデータの転送量を抑えてしまう。その結果、無線ネットワーク環境では TCP 通信のスループットが大幅に減少してしまう。

これらの問題を解決する無線ネットワーク環境に適したトランスポートプロトコルとして、RPLD [2] や TCP-westwood [3] など様々なものが提案されている。それら提案プロトコルの中で、TCP-J [7] は無線ネットワーク環境において従来 TCP との公平性を保ち、スループットを向上させる有効な手法であることが NS2 [6] を用いたシミュレーション評価で確認されている。しかし、実世界のネットワークではシミュレーションのような理想的な環境で通信を行えることは少なく、実装による性能評価は不可欠である。そこで本稿では TCP-J を実装し、実際に TCP 通信を行うことにより評価を行う。

2. TCP-J の設計

TCP-J は SACK (Selective ACKnowledgment) [4] に Bandwidth Estimation 機構 [3] を組み込んだ、End-to-End 型のプロトコルである。TCP-J はセグメント損失の検出時に、Bandwidth Estimation 機構により見積もられたバンド幅を用いて輻輳ウィンド

ウサイズなどを決定することで、不当にデータの転送量を減少させることを防ぐ。また、モバイル端末の移動により接続するネットワークが変わった場合、ネットワークリソースの変化に対応した制御も行っている。TCP-J は送信側のみの修正で実現可能であり、シミュレーションによる性能評価の結果、無線区間のビット誤り率が 4×10^{-6} のとき、NewReno と比べて最大約 191 % のスループット向上が認められている。

TCP-J の Bandwidth Estimation 機構について簡単に説明する。Bandwidth Estimation 機構とは、ACK の到着間隔、セグメントサイズから、現在有効なバンド幅を見積もる機構である。送信側で ACK を受信した時刻を t_k 、その ACK に対応するデータ量を d_k とすると、TCP-J では、(1) 式のようにバンド幅のサンプル BWE_k^{sample} を見積もる。 t_{k-1} は 1 つ前の ACK の到着時刻である。

$$BWE_k^{sample} = \frac{d_k}{t_k - t_{k-1}} \quad (1)$$

TCP-J では、算出した BWE_k^{sample} を (2) 式で平滑化することによって有効なバンド幅 BWE_k を得る。 α は平滑化係数を表している。

$$BWE_k = \alpha \times BWE_{k-1} + (1 - \alpha) \times \frac{BWE_k^{sample} + BWE_{k-1}^{sample}}{2} \quad (2)$$

TCP-J では輻輳制御の際、(2) 式で求められた BWE_k と RTT の最小値 RTT_{min} の積をスロースタート閾値や輻輳ウィンドウサイズに用いることで、無線区間におけるセグメント損失による不必要なスループットの低下を抑制する。

3. TCP-J の実装と改良

本稿では、TCP-J を UNIX 系のオペレーティングシステムである NetBSD1.6.1-Release のカーネルに実装した。NetBSD1.6.1-Release に実装されている TCP のバージョンは Reno と NewReno があり、SACK は実装されていない。TCP-J は SACK をベースとしたプロトコルであるので、まず SACK の実装がされている OpenBSD3.2-Release から NetBSD に SACK の移植を行った。その後、TCP-J の Bandwidth Estimation 機構を追加し、推測したバンド幅を用いた輻輳制御の機能を実装した。ハンドオフ時の輻輳制御の部分は本稿では実装を行わず、今後の課題とした。

3.1 TCP-J の改良

文献 [7] で提案した TCP-J を NetBSD に実装し、

実際に通信を行った結果、いくつかの不具合が発見された。そこで、本稿ではTCP-Jの設計に以下の3点について変更を行った。

1つ目は、3つの重複ACKによるセグメント再送後の輻輳ウィンドウサイズの設定である。TCP-Jではセグメント損失を検出した場合、Bandwidth Estimation 機構から求められたバンド幅を用いて輻輳ウィンドウサイズなどを決定し、無線環境において不当に転送速度を下げることを防ぐ。しかし、予想に反しRenoなどの従来TCPよりも転送速度を下げってしまう場合がある。そこで、そのような場合には従来TCPで用いられるウィンドウサイズの方を採用するように変更を行った。

また、ACKの到着間隔を平滑化するように変更を加えた。TCP-Jではバンド幅を推測する際にACKの到着間隔を利用するが、この到着間隔に大きな揺らぎがあり、算出されたバンド幅にもバラツキが出た。そこでACKの到着間隔を(3)式によって平滑化し、より有効なバンド幅を算出できるように変更を行った。(3)式において、 $s_ack_interv_k$ はACK到着間隔、 s_ack_interv は平滑化されたACK到着間隔、 β は平滑化係数をそれぞれ表している。

$$\left. \begin{aligned} s_ack_interv_k &= \beta \times s_ack_interv_{k-1} + \\ (1 - \beta) \times ack_interv_k \end{aligned} \right\} \quad (3)$$

3つ目の変更はセグメント損失検出後のスロースタート閾値設定のタイミングである。TCP-Jの設計では、高速リカバリフェーズ終了後にスロースタート閾値を設定するが、高速リカバリフェーズにおいては通常バンド幅が減少しているため、スロースタート閾値を必要以上に小さく設定してしまう。そこで、セグメント損失を検出後すぐに、見積もられたバンド幅 BWE を用いてスロースタート閾値を設定するように変更した。

4. TCP-Jの解析／評価

本章では、TCP-Jの実装による性能評価について説明する。Bandwidth Estimation 機構の平滑化係数 α には0.9を用いた。実験に用いたマシンのスペックを表1に示す。

表1 マシンスペック

マシン名	CPU	Memory
Soekris net4501	AMD 134 MHz	64 Mbyte
XSB 2400	XEON 2.4 GHz	512 Mbyte × 2

無線ネットワーク環境は雑音や干渉など様々な要

因でネットワークの状態が変化してしまうため、評価を行うのが難しい。そこで本稿では、有線ネットワークを構築し、中間ルータで帯域制御を行う手法を用いた。ルータでの帯域制御にはdummynet[5]を利用した。dummynetは、FreeBSD OSに組み込まれている帯域制御とプロトコルテストを行うためのツールである。dummynetはマシンを通過するセグメントをインターセプトし、伝送遅延やセグメント損失などの効果をエミュレートしたパイプに通すことによって、様々なネットワークを実現することができる。

dummynetによる帯域制御によって無線環境をエミュレートしたネットワークにおいて、TCP-Jと従来TCP(Reno, NewReno, SACK)との通信性能の比較を行った。実験においてSenderとReceiverにはNetBSD1.6.1-Releaseを、RouterにはFreeBSD5.1-Releaseをそれぞれインストールし測定した。各実験に共通なネットワークの基本的なパラメータとして、最大セグメントサイズを1460 bytes、最大ウィンドウサイズを65535 bytes、Routerのキューサイズを30 segments、遅延確認応答を200 msecとした。

本章において、4.2.1章、4.2.2章および4.3章は文献[7]におけるシミュレーション評価で行われた実験であり、4.1章および4.2.3章は本稿で新たに行った実験である。

4.1 Bandwidth Estimation 機構の評価

TCP-JのBandwidth Estimation 機構の妥当性について評価する。まずACK到着間隔の平滑化に用いる係数 β の最適値について検討する。実験には図1に示す単純なトポロジのネットワークを用いて、dummynetにより帯域制御を行った。実験には表1のXSBマシンを用いた。

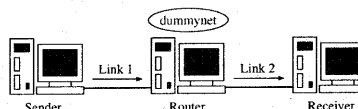


図1 実験トポロジ1

表2 実験トポロジ1におけるリンクパラメータ1

リンク名	バンド幅	伝播遅延
Link 1	10 Mbps	45 msec
Link 2	2 Mbps	0.02 msec

リンクのパラメータは、dummynetにより表2のように設定した。平滑化係数 β を0から0.9まで変

化させ測定した。

送信側で算出された見積もりバンド幅 BWE を出力し、それらの平均値と標準偏差を計算した。その結果を図2に示す。

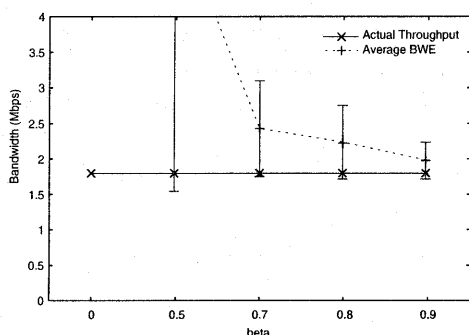


図2 平滑化係数 β と BWE の関係

図2が示すように、ACK 到着間隔の平滑化を行わない場合 ($\beta = 0$) や β の値が0.5以下の場合、実際よりも非常に大きくバンド幅を見積もってしまう。これは、dummysnetによる帯域制御の影響によりACKの到着間隔に大きなバラツキが出るためであり、有効なバンド幅を見積もるためには β は0.9前後の値を取る必要がある。そこで、本稿では β の値を0.9として実験した。

4.2 従来TCPとのスループットの比較

無線リンクでは、雑音や干渉などにより発生するランダムなセグメント損失と、ハンドオフ時などにより一定時間通信が不通になった場合に発生するバースト的なセグメント損失があるので、それぞれのセグメント損失をdummysnetによりエミュレートし、実験した。

4.2.1 ランダムなセグメント損失に対する評価

実験には図1に示すネットワークを用いて、dummysnetにより帯域制御を行った。まずランダムなセグメント損失に対する評価について説明する。

Link 1を有線リンク、Link 2を無線リンクと想定してdummysnetによる帯域制御を行った(表2)。Link 2において、セグメント損失率を 1×10^{-4} から 1×10^{-1} まで変化させた。

図3は、ランダムにセグメント損失を起こした場合のReno, NewReno, SACK, TCP-Jの各スループットを示している。図3においてx軸はセグメント損失率、y軸はスループット (Mbit/sec) を表している。図3から分かるように、NewRenoやSACKはRenoと比べて若干スループットを改善する場合もあるが、やや不安定である。しかし、TCP-Jは安

定して従来TCPよりもスループットを大幅に改善し、特にセグメントの損失率が 1×10^{-2} 以上になった場合に他のTCPよりも大きなスループットを得られることが示された。無線区間でのランダムなセグメント損失率が 2×10^{-2} のとき、NewRenoと比べて約128%, SACKと比べて約110%のスループット向上が認められた。

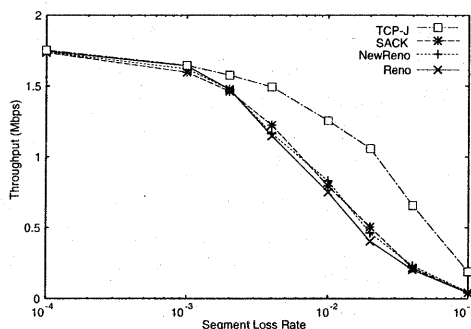


図3 ランダムなセグメント損失が生じた際の各TCPのスループット

4.2.2 バースト的なセグメント損失に対する評価

次にバースト的なセグメント損失に対する評価について説明する。実験には、ランダムなセグメント損失に対する評価の場合と同様、図1に示すネットワークを用いて、dummysnetによりバースト的なセグメント損失を発生させた。通常のdummysnetではランダムなセグメント損失しか発生させることができないため、dummysnetのソースコードに若干の変更を加えてバースト的なセグメント損失を実現した。

Link 1を有線リンク、Link 2を無線リンクと想定してdummysnetによる帯域制御を行った(表2)。連続したデータ送信を行っている最中に、一定間隔(5 sec)ごとにLink 2を不通にさせてバースト的なセグメント損失を起こした。Link 2における通信の不通時間は1~120 msecまで変化させた。Link 2におけるランダムなセグメント損失率は、各TCP間でスループットの差がない 1×10^{-4} とした。

図4にバースト的にセグメント損失を起こした場合のReno, NewReno, SACK, TCP-Jの各スループットを示す。図4において、x軸はLink 2における5 sec毎の通信の不通時間、y軸はスループット (Mbit/sec) を表している。図4に示すように、NewRenoはバースト的なセグメント損失に対し、RenoやSACKよりも若干高いスループットを取っているが、あまり大きな差はない。しかし、TCP-Jは従来TCPに比べて全体的に大きくスループットを改善し、特に通

信の不通時間が 40 msec のとき、NewReno と比べて約 77 %, SACK と比べて約 80 % のスループット向上が認められた。

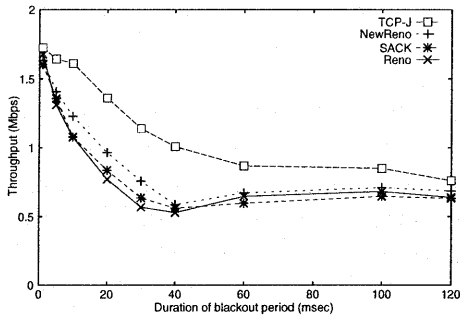


図 4 バースト的なセグメント損失が生じた際の各 TCP のスループット

4.2.3 有線区間にボトルネックリンクが存在する場合の評価

通常、無線ネットワーク環境においては無線区間がボトルネックリンクである場合が多い。しかし、有線ネットワークの混雑状況によっては有線区間がボトルネックリンクとなることも考えられる。そのような場合に TCP-J がどのような挙動を示すか調べる。

実験には図 1 のネットワークを用いて、Link 1 を有線区間、Link 2 を無線区間として dummynet による帯域制御を行った。表 3 に示すように、有線部分の遅延を増加させ、帯域を小さくすることにより、ネットワークの輻輳状態をエミュレートした。また、Router のキューサイズも 5 segments と小さく設定した。

表 3 実験トポロジ 1 におけるリンクパラメータ

リンク名	バンド幅	伝播遅延	セグメント損失率
Link 1	500 Kbps	60 msec	0
Link 2	2 Mbps	0.02 msec	$1 \times 10^{-4} \sim 1 \times 10^{-1}$

図 5 に、有線区間の輻輳状態をエミュレートした環境での各 TCP のスループットを示す。図 5 において、x 軸は Link 2 におけるセグメント損失率、y 軸はスループット (Kbit/sec) を表している。図 5 が示すように、TCP-J はスループットが NewReno や Reno よりもやや低く、SACK よりも若干高い値を取っており、従来 TCP と比べてスループットの面で大きな優劣を持たない。これは、ネットワークの帯域が狭いために TCP-J の見積もりバンド幅 *BWE* も小さな値を取り、輻輳ウィンドウサイズを小さく設定す

るためにスループットがあまり増加しないのだと考えられる。また、Reno などの従来 TCP よりも小さく輻輳ウィンドウを設定しないように設計に変更を加えたため、転送速度を下げ過ぎることもない。以上から、TCP-J はネットワークが輻輳状態にある場合には従来 TCP と同様に輻輳ウィンドウの値を下げ、輻輳回避を行うことができるといえる。

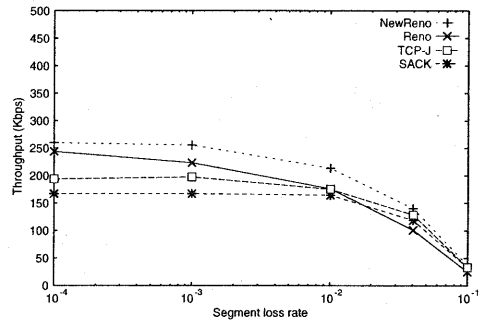


図 5 有線区間が輻輳状態である場合の各 TCP のスループット

4.3 従来 TCP との公平性の評価

最新の Window や Linux などのオペレーティングシステムには NewReno や SACK が実装されているので、NewReno、SACK と TCP-J との間の公平性について評価を行う。実験には図 6 に示すネットワークを用いた。図 6 において、Sender 1 は Receiver 1 と NewReno による通信を、Sender 2 は Receiver 2 と SACK による通信を、Sender 3 は Receiver 3 と TCP-J による通信をそれぞれ行った。実験マシンには、Senders と Receivers に表 1 における Soekris マシン、Routers に XSB マシンを用いた。

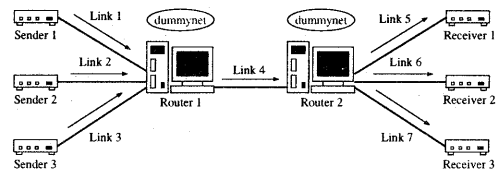


図 6 実験トポロジ 2

表 4 実験トポロジ 2 におけるリンクパラメータ

リンク名	バンド幅	伝播遅延	セグメント損失率
Link 1~3	3 Mbps	10 msec	0
Link 4	3 Mbps	20 msec	0
Link 5~7	3 Mbps	10 msec	$0 \sim 2 \times 10^{-2}$

図 6 のネットワークにおけるリンクの各パラメー

タを表4に示す。Link 5～7のセグメント損失率は0, 2×10^{-3} , 2×10^{-2} と変化させた。

図7にセグメント損失率が0, 2×10^{-3} , 2×10^{-2} の場合のNewReno, SACK, TCP-Jの平均スループットを示す。図7のx軸はセグメント損失率, y軸はスループット (Mbit/sec) を表している。

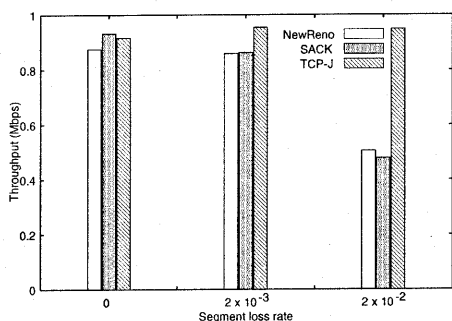


図7 NewReno, SACK, TCP-J 間の公平性 (平均スループット)

図7が示すように、セグメント損失率が0の場合、TCP-JはNewRenoとSACKの通信を圧迫せず、ほぼ同様のスループットで通信を行っている。しかし、セグメント損失率が 2×10^{-3} の場合、TCP-JはNewRenoやSACKに比べて若干高いスループットを示し、さらに高いセグメント損失率である 2×10^{-2} の場合には、従来TCPよりも大幅に高いスループットを示している。これは、TCP-Jが無線ネットワーク環境に適応しており、不必要なスループットの低下を抑制し、現在利用可能なバンド幅を有効に利用したためであると思われる。

しかし、高いセグメント損失率をもつネットワークにおいて、TCP-Jは他のTCP通信を不当に圧迫している可能性がある。そこで、図6のネットワークにおいてセグメント損失率を 2×10^{-2} に設定し、NewRenoのみによる通信とSACKのみによる通信をそれぞれ行った。その結果を表5に示す。

表5 従来TCPのみによる平均スループットとの比較

	同一TCP	混在	スループット減少率
NewReno	0.5060	0.4975	-1.7 %
SACK	0.4783	0.4832	1.0 %

表5から分かるように、NewRenoやSACKはTCP-Jの存在の有無によってスループットが変動しておらず、ほぼ同様の値を取っている。つまり、図7におけるNewRenoとSACKのスループット低下

は、TCP-Jとリンクを共有したことによって起こったのではなく、セグメント損失率の増加に伴うものであることが分かる。以上のことから、TCP-Jは従来TCPの通信を圧迫することなく、無線リンク上のバンド幅を有効に利用することができるといえる。

5. 結 論

本稿では、NetBSD1.6.1-Releaseのカーネル内に無線ネットワーク環境に適したトランスポートプロトコルTCP-Jを実装し、dummynetにより無線ネットワーク環境をエミュレートして既存のTCPとの比較を行った。その結果、TCP-Jは無線リンクでのランダムなセグメント損失率が 2×10^{-2} のとき、NewRenoと比べて約128%, SACKと比べて約110%のスループット向上を得ることができた。無線リンクにおいて、5 secごとに40 msecの間通信が不通になりバースト的なセグメント損失が生じた場合でも、NewRenoと比べて約77%, SACKと比べて約80%程度、通信効率を改善することができた。また、TCP-Jはセグメント損失率が0の場合は従来TCPと公平にバンド幅を分け合い、セグメント損失率が増加し従来TCPのスループット低下によってリンク上のバンド幅に空きができた場合には、他のTCP通信を圧迫せず、空いた帯域を有効利用してスループットを改善することが分かった。

今後の課題として、広域環境での実験やソースコードの配布、他の無線環境に適応したプロトコルとの比較などが挙げられる。

文 献

- [1] H. Elaarag. "Improving TCP Performance over Mobile Networks," ACM Computing Surveys, Vol.34, No.3, pp.347-374, Sep. 2002.
- [2] Nihal K. G. Samaraweera and G. Fairhurst. "Reinforcement of TCP Error Recovery for Wireless Communication," ACM SIGCOMM Computer Communication Review, Vol.28, Issue 2, pp.30-38, Apr. 1998.
- [3] S. Mascolo, C. Casetti, M. Gerla, S. S. Lee, and M. Sanadini. "TCP Westwood: Bandwidth Estimation for Enhanced Transport over Wireless Links," In Proceedings of ACM Mobicom 2001, pp.287-297, Jul. 2001.
- [4] M. Mathis and S. Floyd and A. Romanow. "TCP Selective Acknowledgement Options," Internet RFC2018, IETF, Oct. 1996.
- [5] The Network Emulator for FreeBSD - dummynet. <http://info.iet.unipi.it/luigi/ip.dummynet/>.
- [6] The Network Simulator - ns-2. <http://www.isi.edu/nsnam/ns/>.
- [7] 佐藤徳彦, 國司光宣, 寺岡文男, "TCP-J: 無線ネットワーク環境に適したトランスポートプロトコル," 情報処理学会論文誌, Vol.43, No.12, pp.3848-3858, Dec. 2002.