無線LAN通信における周辺端末の情報に基づく輻輳ウィンドウサイズ決定手法の評価

飯尾 明日香[1]　小口 正人[1]

概要：現在，自動車など数百種類の高性能センサが搭載された端末は，センサ情報から精密な周囲の環境情報を取得できる．しかし，通常，WLAN（Wireless Local Area Network）において用いられる通信パラメータは各々が独自のアルゴリズムに従って設定されているため，その場の環境に応じた通信設定は行われない．したがって，センサから環境情報を取得可能な端末であってもリソースを効率的に利用した通信は行われておらず，必ずしも最適ではない状態で通信を行っているという問題がある．そこで，本研究では，そのような端末が，取得した環境情報を無線通信パラメータ設定に利用し，各端末がそれぞれの状況に合った最適な通信設定を行う手法の検討を行う．上位層である TCP（Transmission Control Protocol）の輻輳制御に着目し，端末の周辺情報として，周辺端末数および端末の移動速度を利用することで適する輻輳ウィンドウサイズを決定し，スループットおよび端末間の公平性の向上を目指す．

An Evaluation of Congestion Window Size Setup Technique Based on the Context in Wireless LAN

Asuka Iio[1]　Masato Oguchi[1]

1. はじめに

現在，自動車には数百種類のセンサが搭載されており，自動車はこれらのセンサから GPS（全地球測位システム）による車両位置を始め，車両間隔，スピード，周辺端末数，加速度など数多くの情報を取得できる．また，車載センサはドライバの安全機能強化の面から高精度化が進んでおり，搭載数も年々増加の傾向にある．このようなセンサから得た周囲の環境情報を Context[1] と呼ぶが，この Context を利用した Context-Aware Computing[2] は，近年，システム連携の仕組みが発展してきたことから実現が期待され，重要性が高まっており，自動車の場合には事故防止や車両安定制御等に役立てられている．

一方，近年 Wireless Local Area Network（WLAN）を用いた通信が一般化しているが，WLAN において通信に用いられる通信パラメータは各々が独自のアルゴリズムに従って設定されており，上位層ではその場の環境に応じた通信設定は行われない．したがって，センサから環境情報を取得可能な端末であってもリソースを効率的に利用した通信が行われておらず，必ずしも最適ではない状態で通信を行っているという問題がある．そこで，本研究では，環境情報を無線通信パラメータ設定に利用し，各端末が周囲の状況に応じた最適な設定を行うことで，通信効率を向上させる手法の検討を行う．

本研究では，環境情報の利用先として上位層の Transmission Control Protocol(TCP）に着目した．TCP において，確認応答（ACK）を受けずに送信できるパケットの最大数であるウィンドウサイズのうち，輻輳ウィンドウサイズ（cwnd）は輻輳制御アルゴリズムによりエンドツーエンドで制御されているため，各端末は必ずしも周囲の状況に合った通信を行っているとは言えない．そこで，Context として周辺端末数を利用し，各端末が最適な cwnd を用いて通信を行うことで通信性能の向上を目指す．

本研究では，輻輳制御アルゴリズムにより cwnd が制御される一般的な TCP を用いて通信を行った場合と，周辺端末数に応じ最適な cwnd 値に固定して通信を行う独自の TCP を用いて通信を行った場合のそれぞれにおいて，周辺

[1] お茶の水女子大学
Ochanomizu University

端末数に伴うスループットの変化をシミュレーションにより測定し，評価を行う．ここで，提案する独自の TCP は，帯域幅と往復遅延時間（RTT）の乗算で表される帯域幅遅延積を周辺端末数に応じて均等に分け cwnd を決定し，その値に固定したまま通信を行う仕様となっている．これにより，周辺端末数という Context を活用した公平性の高い通信の実現を目指す．

本稿では，AP からの距離に応じてデータレートが異なるより実環境に近いモデルを使用し，提案手法の評価を行う．また端末の移動性を考慮したモデルを検討し，その評価を行う．

なお，本研究の実験にはネットワークシミュレータ OMNeT++ version4.1[3] を使用する．OMNeT++のプラットフォーム上で，INET Framework[4] を拡張したシミュレーションモデルである INETMANET[5] を動作させ，実験を行った．

2. TCP

2.1 TCP による輻輳制御

WLAN の国際標準として IEEE 802.11 が規定されており，IEEE 802.11 の上位層においては TCP により輻輳制御が実現されている．輻輳とはネットワークの混雑を表す言葉であり，ネットワーク上で多量のトラフィックが発生して，通常の通信が困難になることをいう．TCP は，輻輳制御により輻輳を回避する仕様となっている．代表的な TCP として Tahoe，Reno，NewReno，Vegas，Cubic 等があり，輻輳制御アルゴリズムは TCP ごとに様々な方式となっている．

TCP では，ウィンドウフロー制御を実行することによって，輻輳の防止・早期回避を可能にしているが，一方で，輻輳回避フェーズにおける輻輳による cwnd の減少幅が，その後の線形増加時の増加幅と比較し大きいため，一度輻輳が起き cwnd が下がると，cwnd が増加するまでに時間がかかり，輻輳制御が原因となったスループットの低下が起こる場合がある．以下では，TCP Tahoe と TCP Reno の輻輳アルゴリズムについて説明し，本稿で比較評価に用いた TCP である TCP NewReno について触れる．

2.2 TCP Tahoe

TCP Tahoe は，スロースタートフェーズと呼ばれる，ACK セグメントを受信した分だけ cwnd を増加させる動作を行う (式 (1)，if 条件式)．

$$cwnd \longleftarrow \begin{cases} cwnd + 1 & \text{if cwnd} < \text{ssthresh} \\ cwnd + \frac{1}{cwnd} & \text{otherwise} \end{cases} \quad (1)$$

cwnd=1 でセグメントを送信すると，受信側から 1 つの ACK セグメントが送信されるため cwnd は 2 となる．cwnd が 2 のとき 2 つのセグメントを送信するため，受信側からは 2 つの ACK セグメントが送信され，cwnd は 4 となる．このようにスロースタートフェーズでは cwnd を 1,2,4,8,16,…と指数関数的に増加させていく．ここで，ssthresh はスロースタートから輻輳回避のフェーズへ移行する際の閾値である．ssthresh 初期値は多くの実装で広告ウィンドウサイズが設定されるが，ある程度大きい値であれば任意である．

また，輻輳が起こると ssthresh は以下のように更新される．

$$ssthresh \longleftarrow \frac{ssthresh}{2} \quad (2)$$

その後，cwnd を 1 まで下げ，再びスロースタートフェーズを行う．やがて cwnd が ssthreth に達すると輻輳回避フェーズへ移行し，cwnd を式 (1) の otherwise 条件式を用いて線形的に増加させる．ここで再度輻輳が起こると，再び式 (2) を用いて ssthreth を更新し，cwnd を 1 まで下げ，スロースタートフェーズへ移行する．TCP Tahoe はこの動作を繰り返す（図 1）．また，高速再送アルゴリズムが採用された点も TCP Tahoe の特徴である．

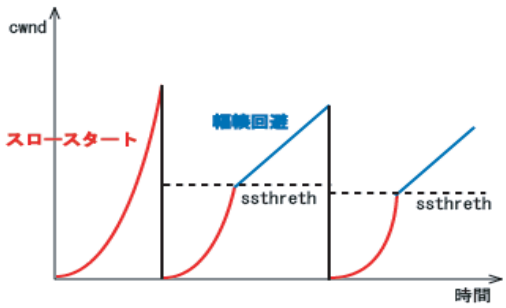

図 1 TCP Tahoe における cwnd の変化

2.3 TCP Reno

TCP Reno は TCP Tahoe をベースとしたアルゴリズムで，TCP Tahoe に対してエラーが起きても必ずしも cwnd を 1 まで下げないように改良したものである．すなわち，始めはスロースタートフェーズで cwnd の急速な増加を図り，その後，cwnd が ssthresh に到達すると，輻輳回避フェーズへと移行する．輻輳回避フェーズでは，輻輳が起こると ssthresh を半分に更新し，cwnd を ssthresh まで下げてから線形的に増加させ，輻輳が起こる度にこれを繰り返す (図 2)．また，パケットロスなどによりタイムアウトが起きると，cwnd を 1 に変更しスロースタートフェーズの動作へと戻る．

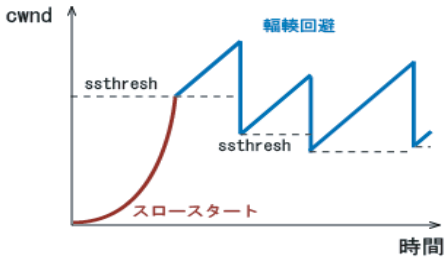

図 2 TCP Reno における cwnd の変化

2.4 TCP NewReno

TCP Reno では SACK（selective acknowledgement）がない場合に，複数パケットが損失すると高速再転送および高速リカバリアルゴリズムが有効に動作しない問題があった．この高速リカバリアルゴリズムを修整したものが TCP NewReno である．

2.5 独自の TCP

2.2〜2.4 節で述べたように，通常，TCP では cwnd を輻輳制御アルゴリズムにより動的に制御している．しかし，本研究では Context として周辺端末数を利用し，各端末が初めから適切な cwnd 値に固定して通信を行うことで，通信性能の向上を試みる．そこで，文献 [6] より，帯域幅遅延積（式（3）上式）を用い，これを端末数に応じて均等に分けることで，公平かつ効率の良い通信を目指す．

$$\begin{cases} \text{帯域幅遅延積}\ [bit] = \text{帯域幅}\ [bit/s] \times RTT[s] \\ cwnd = \text{帯域幅遅延積} \div \text{端末数} \end{cases} \tag{3}$$

ここで，Round Trip Time（RTT）は往復遅延時間を表す．

本稿では，すべての端末数において式（3）を適用し，端末数ごとに適切な cwnd を定めて通信を行う TCP を独自の TCP と呼ぶ．

3. 検証実験

3.1 概要

実験には，AP（Access Point）から等距離にある複数の端末が，AP と 2 台のルータを経由しサーバにパケットを送信するシンプルなシミュレーションモデルを使用した (図 3).

このモデルにおいて，周辺端末数を用い，2.5 節で述べた式 (3) を適用して cwnd を定めて通信する独自の TCP と，その比較対象として，一般的に使用されている TCP である TCP NewReno の 2 種類の TCP を使用して，スループットを測定した．ここで，端末は，AP を中心とした円周上に均等に配置しているものとする．

また，無線規格は IEEE802.11g を使用し，広告ウィンドウサイズは十分大きい値に設定した．シミュレーションパラメータを表 1 に示す．なお，表 1 における RTT とは，OMNeT++上でパラメータとして設定するコネクションのみの往復遅延時間を表した時間であり，実際の測定値とは異なる．

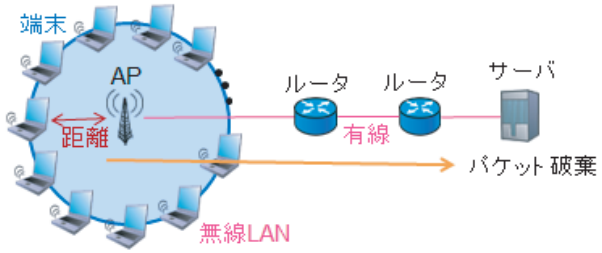

図 3 シミュレーションモデル

表 1 シミュレーションパラメータ

無線規格	IEEE8002.11g
シミュレーション時間	20sec
RTT	40msec
周波帯域	2.4GHz
有線部分の伝送レート	54Mbps
最大セグメントサイズ	65,280Byte
広告ウィンドウサイズ	33,423,360Byte
端末数	1〜50 台

端末の伝送レートの設定方法はいくつかあるが，本研究では一般的な通信で良く使われている，距離に応じた伝送レートの設定方法を用いることとした．距離と伝送レートの相関関係については，無線機器あるいはベンダごとで異なるが，本研究では表 2 に示す Cisco 社の Aironet 1130AG IEEE 802.11a/b/g AP のパラメータを利用する [7].

表 2 Aironet 1130AG IEEE 802.11a/b/g AP　伝送レート設定値

AP との距離（m）	10	20	30	90	120	150	200	250
伝送レート（Mbps）	54	48	36	24	18	12	9	6

3.2 端末の静止状態における性能比較

まず，図 3 のシミュレーションモデルにおいて，全ての端末が静止状態にある場合の性能を測定した．TCP NewReno，独自の TCP のトータルスループットをそれぞれ図 4，図 5 に示す．

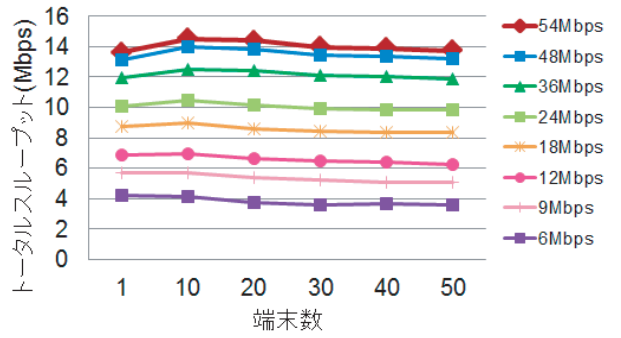

図 4　TCP NewReno の静止時トータルスループット

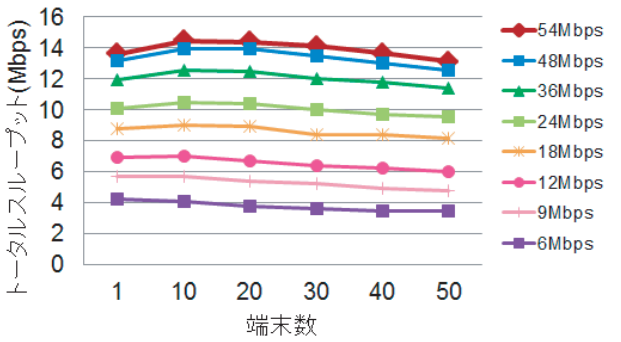

図 5　独自の TCP の静止時トータルスループット

次に，実験に用いた 2 種類の TCP について，伝送レート 6Mbps，端末数 50 台で通信を行った場合の各端末のそれぞれのスループットを図 6，図 7 に示す．

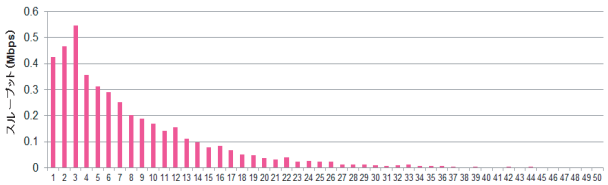

図 6　NewReno における端末数 50 台の静止時スループット

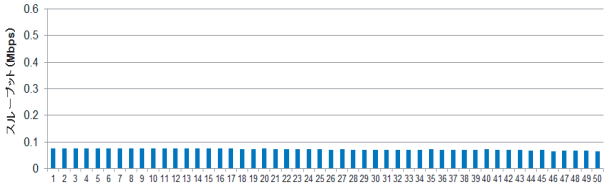

図 7　独自の TCP における端末数 50 台の静止時スループット

最後に，MAC 層における再送回数を図 8 に示す．本実験では 1 つのパケットにおける再送の上限値を 7 回に設定し，シミュレーションを行った．

図 4，図 5 により，TCP NewReno と独自の TCP のトータルスループットを比較すると，多少の差はあるがほぼ同じ程度である．しかし，図 6，図 7 を見るとわかるように，TCP NewReno は端末によってスループットが著しく異なっており，一部の端末が帯域を占有して，他の端末は帯域をほとんど使えない状態となっている．したがって，独自の TCP は TCP NewReno に比べて公平性が非常に高くなっている．

なお，図 8 より，MAC 層における再送回数は，独自の TCP が TCP NewReno に比べて多くなっている．これは，TCP NewReno が一部の端末のみ通信が行えていることによると考えられるが，今後さらに詳細な解析が必要である．

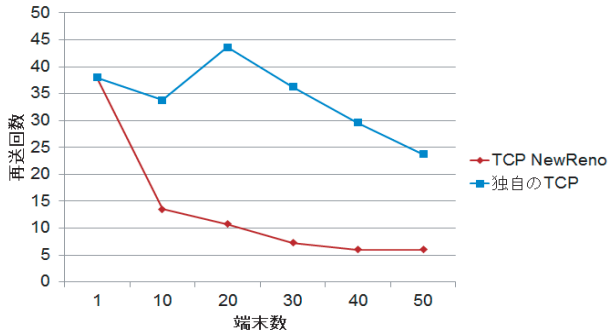

図 8　静止時の再送回数比較

3.3　端末の移動性を考慮した性能比較

続いて，図 3 において，全ての端末が，配置された円周上を，時計回りに移動したまま通信を行うモデルについて検討した．本研究では，自動車が移動しながら通信を行う状況を想定しているため，端末の速さは 10mps とし，端末数が多い場合でも端末間に最も距離を保つことのできる伝送レート 6Mbps の環境において測定を行った．

静止時とは振舞いが異なるため，特に，特徴的であったシミュレーション結果として図 9，図 10 を示す．図 9 は，独自の TCP で式（3）における RTT 値を 150msec に設定した場合の結果であり，また，図 10 は RTT 値を 170msec とした場合の結果である．

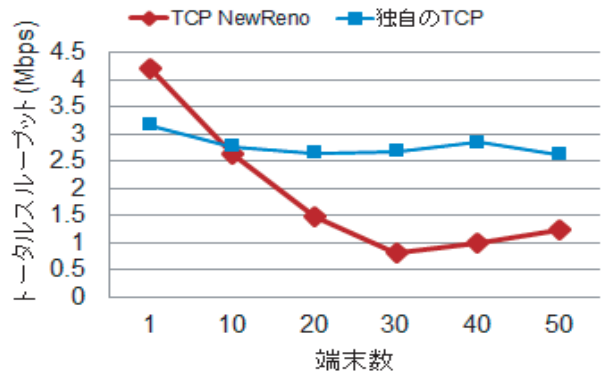

図 9　RTT=150ms に設定した場合のトータルスループット

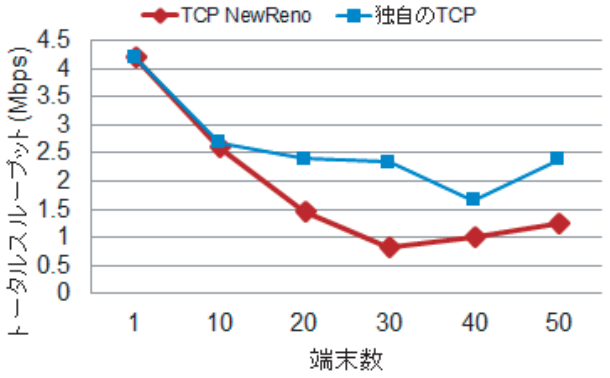

図 10 RTT=170ms に設定した場合のトータルスループット

次に，独自の TCP で RTT を 150msec として測定した場合の各端末のスループットを測定し，TCP NewReno と振舞いを比較した．

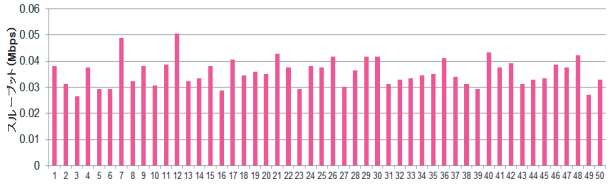

図 11 NewReno における端末数 50 台の移動時スループット

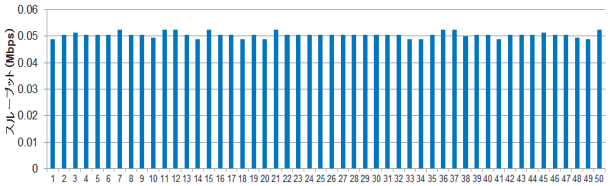

図 12 独自の TCP における端末数 50 台の移動時スループット

最後に，3.2 節と同様に，再送回数を図 13 に示す．

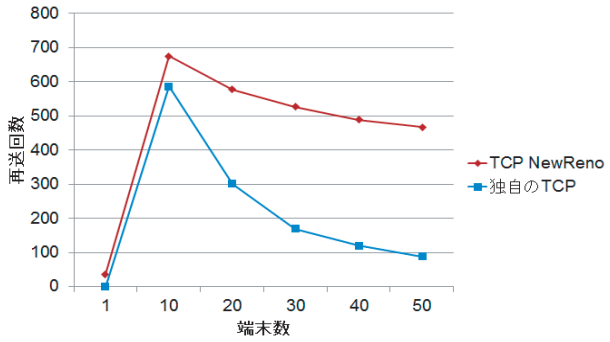

図 13 移動時の再送回数比較

3.4 考察

図 4 と図 5 を比較すると，一般的な TCP である TCP NewReno と，独自の TCP のトータルスループットは同じ程度であった．しかし，図 6，図 7 のように伝送レート 6Mbps，端末数 50 台で通信を行った場合のスループットの内訳を比較すると，TCP NewReno では，帯域を大きく使用している端末と，スループットが 0 に近く，ほとんど通信が行われていない端末が見られ，端末間で不公平が発生していることがわかった．一方で，独自の TCP では各端末が均等に帯域を分け合っているため，スループットに大きな差は見られなかった．よって，トータルスループットがほとんど変わらない場合でも，独自の TCP では従来の TCP よりも公平な通信が行われていることがわかった．

次に，移動性を考慮したモデルにおいて，図 9 と図 10 から，式（3）で設定する RTT の値によって，振舞いが変わることがわかった． RTT を大きく設定，つまり，端末間で分ける帯域幅遅延積を大きく設定すると，端末数が小さいときにスループットが増加するが，一方で，端末数が大きい場合に性能がやや低下するといえる．よって，端末数が大きく変化する環境など，端末数に関わらず一定の通信速度を確保したい場合には，帯域に余裕を持たせるため，式 3 における RTT を大きめに設定するとよい．また，各端末におけるスループットを比較し，端末の静止状態での実験と同様に，移動時も公平性が向上することがわかった．図 13 において再送回数を比較しても，独自の TCP では効率の良い通信が行われていると言える．

以上のことから，提案手法の独自の TCP では，従来の TCP よりも公平性が高く，安定した通信が可能であると言える．

4. おわりに

全ての端末が AP から等距離にあるシンプルなシミュレーションモデルを用い，端末の静止時と移動時のスループットを測定した．この結果から，静止時および移動時において，独自の TCP では，従来の TCP よりも端末間の公平性が向上したことがわかった．また，移動時においては，周囲の状況に応じ最適な cwnd に固定することで，全体のスループットも向上することがわかった．よって，シンプルなモデルを用いた場合，端末が静止している時も，移動している時も提案手法は有用であると言える．

今後は，端末の配置をより複雑化し，実環境に近いモデルを用いて性能を向上する手法を検討したい．

謝辞

本研究を進めるにあたり，株式会社トヨタ IT 開発センターの Onur Altintas 氏，松本真紀子氏に大変有用なアドバイスをいただきました．深く感謝いたします．

参考文献

[1] Day, Anind K. , ”Understanding and Using Context”(2001). Human-Computer Interaction institute. Paper 34. http://repository.cmu.edu/hcii/34

[2] Daniel Salder, Anind K.Dey and Gregory D. Abowd, ”The Context Toolkit : Aiding the Development of Context-Enabled Applications”, In Proccedings of CHI’99, PIttsburgh, PA, May, 1999, ACM Press.

[3] OMNeT++, http://www.omnetpp.org/

[4] INET Framework, http://inet.omnetpp.org/

[5] INETMANET Framework for OMNEST/OMNeT++ 4.x (based on INET Framework), https://github.com/inetmanet/inetmanet/wiki

[6] W.Richard Stevens 著, 橘 康雄, 井上 尚司訳 (2000)『詳解 TCP/IP〈Vol.1〉プロトコル』ピアソンエデュケーション 327pp.

[7] Cisco Aironet 1130AG IEEE 802.11a/b/g AP, http://www.cisco.com/web/JP/product/hs/wireless/airo1130/prodlit/1130ag_ds.html

[8] 松本真紀子, 小口正人, ”無線端末の周辺情報を利用したMAC 層におけるマルチプルアクセス手法の一検討”, 電子情報通信学会ネットワークシステム研究会, NS2011-106, pp.13-18, 2011 年 11 月