Android 異種クライアント端末における公平性を考慮した協調的輻輳制御手法

島田　歩実[†]　　山口　実靖[‡]　　小口　正人[†]

お茶の水女子大学[†]　　工学院大学[‡]

1.　はじめに

近年のロスベース TCP は，より高いスループットを確保するためにアグレッシブな輻輳制御手法を用いているが,無線接続環境においては膨大なパケットが無線 LAN アクセスポイント(-AP)に蓄積され,その結果輻輳が発生するといった問題が生じている．先行研究では,ロスベース TCP の一種である CUBIC-TCP を輻輳制御アルゴリズムとして使用している Android 端末を用いて,無線 LAN (-AP)における ACK パケットの蓄積を回避する輻輳制御手法がスマートフォン端末向けに提案,実装された．本研究では,まずスマートフォン向けの既存手法をタブレット端末にナイーブに適用する．また，RTT 値によってネットワークの負荷を推定することで適切な輻輳制御を行う新たな手法を考案，実装した．性能評価により,通信速度と公平性が先行研究の手法よりもスマートフォンとタブレット共に大幅に向上したことを示す．

2.　先行研究

2.1 カーネルモニタ

カーネル内部の処理は通常バックグラウンドで進められているため，通常ユーザ空間からその処理の様子を監視することはできない．そこで既存研究[1]によりカーネル内部の情報を見るためにカーネルモニタと呼ばれるツールが開発された．カーネルモニタは本研究のベースとして用いられており，Android 端末の TCP 通信時における輻輳ウィンドウサイズ(CWND)や往復遅延時間(RTT)などのカーネル内部の様々なパラメータをリアルタイムにモニタする汎用 PC 向けシステムツールである．

2.2 輻輳制御ミドルウェア

先行研究[2]で開発された輻輳制御ミドルウェアは，カーネルモニタをベースとしたシステムであり，無線 LAN-AP における ACK パケットの蓄積を回避し，複数の端末が同一の AP に接続して通信するときの全体の通信速度と公平性の向上を可能にしている．同一 AP に接続された周辺端末台数と，自端末の RTT 値の 2 つをパラメータとし，適切な CWND 値の最大値と最小値を算出し，補正を行う．周辺端末数は，端末間でブロードキャストを通知し合うことで把握し，自端末の RTT 値はカーネルモニタから取得する．

本システムは，端末の RTT 値が大幅な増加した場合にのみ迅速に CWND を補正するだけであり，その範囲の中でデフォルトの TCP アルゴリズムによる制御が行われる．このように，基本的な TCP の輻輳制御アルゴリズムは変更せずエンドホスト側にのみ変更を加えるだけであるため，本システムの導入は容易である．

3.　本提案手法

3.1 既存研究の問題点

上記既存ミドルウェアにはいくつか欠点がある．まず，協調性の問題である．既存システムは，自端末の RTT 値が大幅に上昇すると，自端末の輻輳制御ミドルウェアの制御を開始する．このように，制御開始前は周辺の状況を把握しておらず，端末各々においてシステムを発動させるため，制御を行っている端末とそうでない端末がネットワーク上に混在することになり，公平性が落ちる場合があった．また,システム発動後も，パラメータとして用いる RTT 値は，自端末の値のみを用いるため，周辺端末の状況と関係なく制御を行うことになる．

二つ目の問題は，CWND の補正値の算出方法である．既存システムは表 1 のように，研究者の経験によって判断された補正値が設定されている．この設定値は，AP における ACK パケットの蓄積を回避するため，RTT 値が増加するにつれ，非常に小さな値になるよう設定されている．しかし，この補正値が小さすぎるために，帯域が使われず無駄になり，結果的に全体の通信速度が著しく低下してしまう要因となることがあった．

表 1　RTT 値をパラメータとした CWND 値算出テーブル

RTT value	0-51	51-105	106-161	161-331	331-
Max_CWND	555	200	100	10	1
MIN_CWND	200	50	10	2	1

3.2 輻輳制御ミドルウェアの改良

本研究では，以上の問題を解決するためにミドルウェアを改良した．

まず，一つ目の協調性の問題を解決するために，本システムの発動タイミングを改良した．先行研究のミドルウェアは，自端末の RTT 値が大幅に上昇すると，自端末の輻輳制御ミドルウェアの制御を開始

A Study of Cooperative Congestion Control Middleware on Android Terminals
† Ayumi Shimada, Masato Oguchi
‡ Saneyasu Yamaguchi
Ochanomizu University (†)
Kogakuin University (‡)

していた．本提案手法では，システム発動前から，TCP 通信を行っている端末全てがブロードキャストによって情報を通知し合い，同一 AP 上の少なくとも一台の端末の RTT 値が大幅に上昇すると，同一ネットワーク上における全端末においてミドルウェアによる制御を開始するように改良した．

また，二つ目の問題を解決するために，Google 社開発の BBR アルゴリズムからヒントを得た．BBR アルゴリズムは，スループットと RTT 値を常に評価することにより，データを送出するペースと伝送の所要時間の関係を把握し，ネットワークが処理可能なペースでデータを送出する．本提案手法は BBR のように処理可能なペースでデータを送出するよう，CWND 値を算出し補正する手法を取る．

図 1 は，後述の実験環境における端末の CWND 値と RTT 値の相関関係を示す．CWND の上限が 600 までは， RTT 値が比例的に上昇するが，それ以降は一定または微小の増減である．つまり，CWND サイズ 600 以降は，パケットがキューから溢れ始めていることが予想される．そこで我々は，CWND 値の上限を，パケットがキューに収まっていると判断できる 600 とし，これを同時接続端末数で分け合うことで，CWND を補正する手法を考案した．

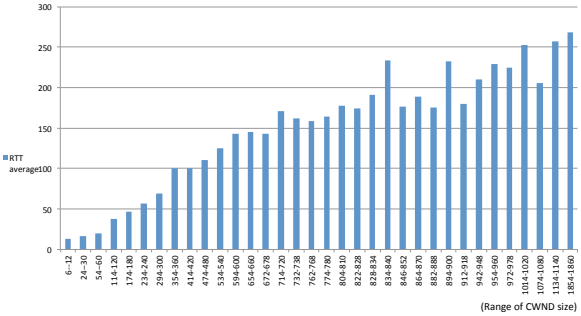

図 1 CWND サイズと RTT 値の関係

4. 本提案手法の評価実験

本提案手法を評価するために，実験を行った．図 2 に実験環境を示す．サーバ機と AP の間に人工遅延装置 dummynet を挟み，有線部の往復遅延時間を特に輻輳が生じやすい高遅延環境を模擬するために 100ms に設定している．この環境において一つの AP に接続し Android 端末とサーバ間において iperf を用いて通信を行い，ネット ワーク全体の通信スループットを測定した．この環境において，Android 端末の台数を 6 台同時通信させた．

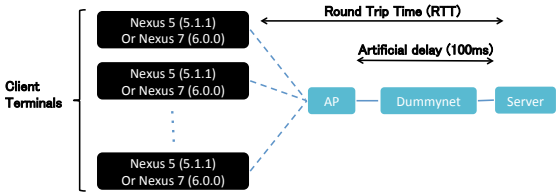

図 2 実験環境

本研究では通信速度と公平性の両方の向上を目指しているため、図 3 図 4 のような散布図で実験結果を評価する．縦軸が公平性，横軸が合計通信速度を示す．図 3 はスマートフォン，図 4 はタブレットの結果である．青い点はミドルウェアを用いずにデフォルト通信を行った結果，赤い点は先行研究のミドルウェアを用いた結果，緑の点は本提案ミドルウェアを用いた結果を示す．それぞれの通信実験を 3 回行ったため，同じ色の点が 3 つずつある．

図 3 の結果に顕著に見られるように，先行研究のミドルウェアは，一部の端末のみにて厳しい輻輳制御がなされ，残りの端末が帯域を自由に使用しているため，合計通信速度は向上する一方で公平性が著しく低下している．しかし，本提案手法の結果は，全ての端末が平等に適切な帯域を分け与えられているため，通信速度と公平性が共に向上している．

図 4 に示すタブレット端末においても同様に，本提案手法の有効性が示された．

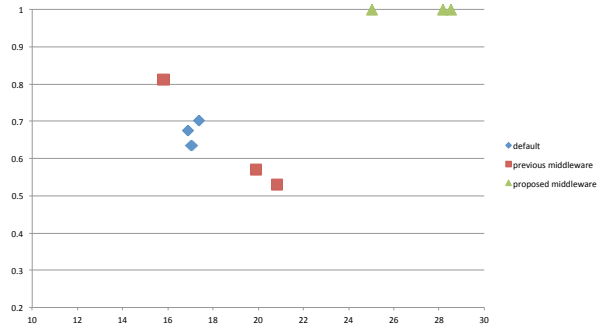

図 3 実験結果(スマートフォン)

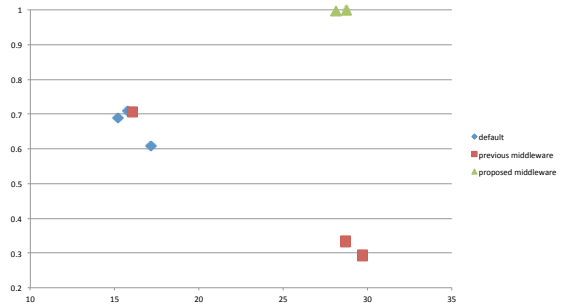

図 4 実験結果(タブレット)

謝辞

本研究は一部，JST CREST JPMJCR1503 の支援を受けたものである．

参考文献

[1] Kaori Miki, Saneyasu Yamaguchi, and Masato Oguchi: “Kernel Monitor of Transport Layer Developed for Android Working on Mobile Phone Terminals,” Proc. ICN2011, pp.297-302, January 2011.

[2] Ai Hayakawa, Saneyasu Yamaguchi, Masato Oguchi: “Reducing the TCP ACK Packet Backlog at the WLAN Access Point,” Proc. ACM IMCOM2015, 5- 4, January 2015.