4ZB-06

ネットワークエミュレータ Mininet による マルチプル Copa フローの性能評価

齋藤 千夏奈 † 岡田 章吾 ‡ 荻野 雅史 ‡ 内海 哲史 †

福島大学 理工学群 共生システム理工学類 † 福島大学大学院 共生システム理工学研究科 ‡

1 はじめに

2009 年頃から，インターネットにおいてバッファリング遅延が増大する現象である Buffebloat [1] が問題視されている．Buffebloat を回避する手段として，バッファリング遅延を抑える輻輳制御アルゴリズム (CCA: Congestion Control Algorithm) である Copa [2] が 2018 年，MIT から発表された．Copa は，伝搬遅延が大きい衛星ネットワーク環境においてもバッファリング遅延を抑え，リアルタイム通信を実現できる CCA として期待される．また Meta (旧 Facebook) 社は，Android 端末への動画配信に Copa を使用している [3]．Copa は Buffebloat を回避する手段として期待されるが，多数の Copa フローがボトルネックリンクを共有する場合の性能についてあまり知られていない．そこで本稿では，多数の Copa フローがボトルネックリンクを共有する場合について，ネットワークエミュレータ Mininet [4] を用いたエミュレーション実験によって，性能評価を行う．

2 輻輳制御アルゴリズム Copa

Copa では，バッファリング遅延の推定値 d_{q} [sec] を用いて，スループットの目標値を $\lambda = \frac{1}{\delta d_{\mathrm{q}}}$ [packets/sec] により決定し，目的関数 $U = \log(\lambda) - \delta \cdot \log(d)$ を最大化することを目指す．ここで，δ はスループットに対して，バッファリング遅延を重み付けするパラメータである．

また，Copa では，δ の値を固定するデフォルトモードと，バッファを埋め尽くすロスベース輻輳制御機構と競合する場合に，δ の値を動的に調整する競合モードが実装されている．デフォルトモードでは，$\delta = 0.5$ を用いる．文献 [2] によると，バッファが直近の過去 $5 \cdot R$ [sec]（本稿で用いる CCP [5] の実装によると，直近の過去 $10 \cdot R$）の間に空である状態を検出した場合，デフォルトモードを継続する．ここで，R は往復遅延時間 (RTT: Round Trip Time) である．また，直近の過去にバッファが空である状態を検出しなかった場合，競合モードに移行する．競合モードでは，パケットの到着やロスに応じて $1/\delta$ を AIMD (Additive-Increase/Multiplicative-Decrease) で変化させる．

3 実験環境

図 1 に，Mininet によるエミュレーション実験で用いるネットワークトポロジを示す．ネットワークは，8 台の送信端末と 8 台の受信端末，2 台の交換機，16 本のアクセスリンク，及び 1 本のボトルネックリンクから構成される．送信端末の CCA として Copa を用いる．ボトルネックリンクの帯域，バッファサイズをそれぞれ 1 [Gbps]，64 [BDP]，端末間の往復伝搬遅延時間を 40 [msec]，実験時間を 120 [sec] とする．ここで，1 [BDP] = 5 [Mbytes] は帯域遅延積を示す．アクセスリンク，ボトルネックリンクはすべて有線リンクを想定し，エラーによるパケットロス率を 0 % とする．すべての送信端末は，iPerf [6] フローを 2，4，8，または 16 本ずつ送信し，合計 16, 32, 64, または 128 本のフローすべてをボトルネックリンクで共有させる．図 1 の送信端末 S_1 から受信端末 R_1 に送信されるフローの 1 つをフロー 1 と呼ぶ．

4 実験結果

図 2 に，フロー数に対する合計のスループットの変化を，図 3 に，フロー数に対するフロー 1 の平均 RTT の変化をそれぞれ示す．図 4 に，フロー 1 のモードの時間変化を，図 5 に，フロー 1 の送信中パケット数の時間変化を，図 6 に，フロー 1 の RTT の時間変化をそれぞれ示す．図 2 より，フロー数に関係なく，Copa フローの合計のスループットは，ボトルネックリンクの帯域である 1 [Gbps] に近い値となっており，高い回線利用率を示すことがわかる．一方，図 3 より，Copa フローの平均 RTT は 64，128 フローのときに増加していることがわかる．これは，フロー数が多い場合に，Copa フローのみでボトルネックリンクを共有しているにもかかわらず，Copa が競合モードで動作していることに起因して

Evaluation of Multiple Copa Flows Using Network Emulator Mininet
Chikana Saito†, Shogo Okada‡, Masashi Ogino‡, Satoshi Utsumi†
*Faculty of Symbiotic Systems Science, Cluster of Science and Technology, Fukushima University
†Graduate School of Symbiotic Systems Science and Technology, Fukushima University

Copyright ©2022 Information Processing Society of Japan.
All Rights Reserved.

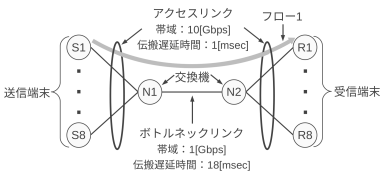


図 1　ネットワークトポロジ

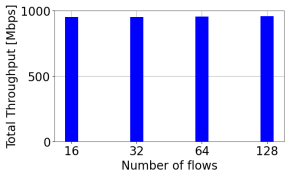


図 2　フロー数に対する合計のスループットの変化

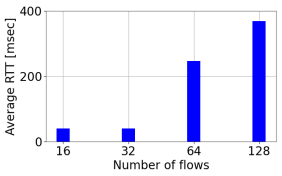
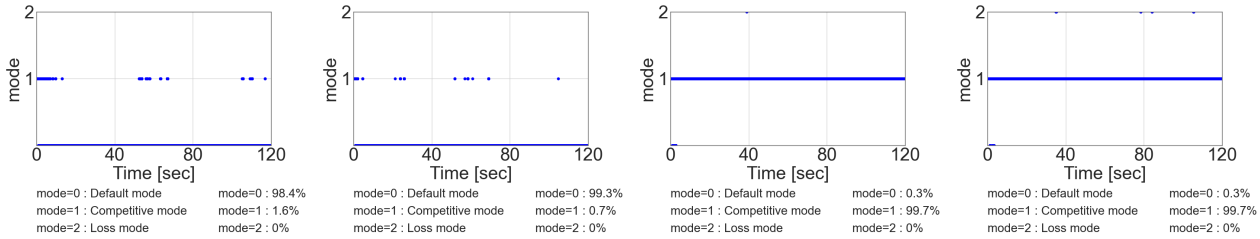
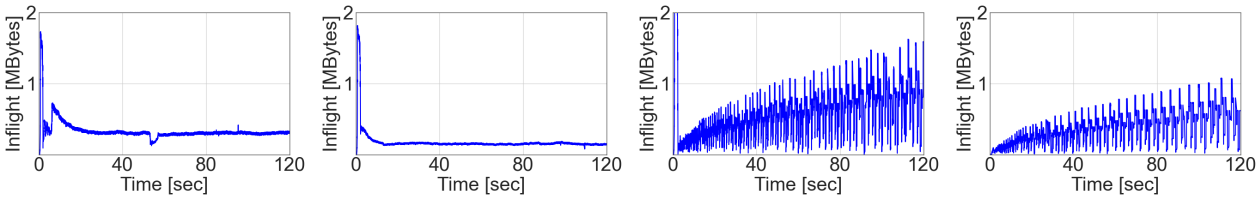


図 3　フロー数に対するフロー 1 の平均 RTT の変化



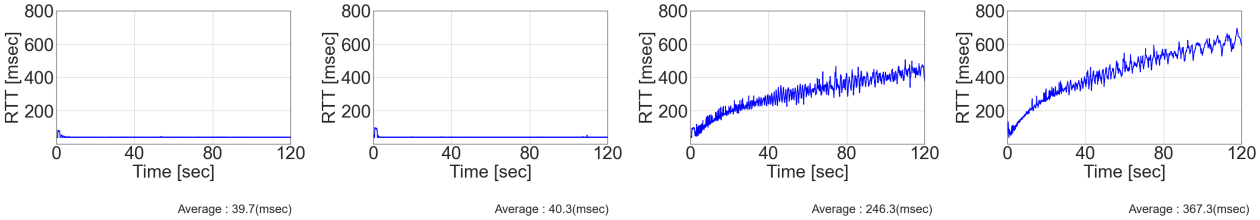
(a) 16 フロー　(b) 32 フロー　(c) 64 フロー　(d) 128 フロー

図 4　フロー 1 のモードの時間変化



(a) 16 フロー　(b) 32 フロー　(c) 64 フロー　(d) 128 フロー

図 5　フロー 1 の送信中パケット数の時間変化



(a) 16 フロー　(b) 32 フロー　(c) 64 フロー　(d) 128 フロー

図 6　フロー 1 の RTT の時間変化

いる．図 4 より，64，128 フローのときに，フロー 1 の Copa が競合モードで動作していることがわかる．また，図 4，5，6 より，Copa が競合モードで動作するとき，送信中パケット数，RTT ともに増加していることがわかる．Copa はフロー数が多い場合，多数の送信パケットでバッファを埋めるため，直近の過去に Copa は最小 RTT を観測できず，競合モードで動作する．Copa が競合モードで動作するとき，AIMD のロスベース CCA と同様に，送信パケットがボトルネックリンクのバッファを積極的に埋めるため，最小 RTT を観測できない状況が継続し，Copa は競合モードで動作し続ける．

5　まとめ

本稿では，多数の Copa フローがボトルネックリンクを共有する場合についての性能評価を行った．多数の Copa フローがボトルネックリンクを共有する場合，RTT が増加する問題を確認した．

今後の課題は，多数の Copa フローがボトルネックリンクを共有する場合においても，バッファリング遅延を抑えることができるように Copa のアルゴリズムを改良することである．

謝辞　本研究は，福島大学学内競争的研究資金 (21RG002) より実施されたものである．

参考文献

[1] J. Gettys. ”Bufferbloat: Dark buffers in the internet”. *IEEE Internet Computing*, 15(3):96–96, 2011.
[2] V. Arun and H. Balakrishnan. ”Copa: Practical delay-based congestion control for the internet”. In *15th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 18)*, pages 329–342, 2018.
[3] N. Garg. ”Copa congestion control for video performance - Facebook Engineering”. `https://engineering.fb.com/2019/11/17/video-engineering/copa/`. (Accessed on 09 Dec. 2021).
[4] Mininet: An Instant Virtual Network on Your Laptop (or Other PC) - Mininet. `http://mininet.org/`. (Accessed on 09 Dec. 2021).
[5] CCP: Congestion Control Plane. `https://ccp-project.github.io/`. (Accessed on 23 Dec. 2021).
[6] iPerf - The ultimate speed test tool for TCP, UDP and SCTP. `https://iperf.fr/`. (Accessed on 03 Jan. 2022).

Copyright ©2022 Information Processing Society of Japan.
All Rights Reserved.