RCCF: Accelerating the Redeployment of Congestion Control Algorithms to Provide Near-Optimal Service for Different Flows

Bo Su, Xianliang Jiang, Guang Jin, Haiming Chen
Ningbo University

ABSTRACT

The heterogeneity of networks continues to increase, and TCP variants emerge in large numbers. Even the TCP algorithm aiming to work under heterogeneous networks cannot perform well and each TCP algorithm has its merit and demerit in specific scenarios. Furthermore, the novel TCP algorithms cannot be evaluated and deployed well in last decade years. In this paper, we propose a *Reconstructed Congestion Control Framework (RCCF)*, which is a novel framework to accelerate the redeployment of congestion control algorithms and provide near-optimal service for different flows. RCCF can evaluate congestion control algorithms sufficiently, deploy them quickly and maximize the advantages of them. We implement RCCF in Linux and the preliminary experiments show its effectiveness.

KEYWORDS

TCP, heterogeneous networks, congestion control framework

1 INTRODUCTION

The edge networks combined by the IEEE 802.11 wireless LAN, the IEEE 802.16 wireless metropolitan area network, 4G and incoming 5G are growing more heterogeneous. The characteristics of their path are diverse and dynamic. Now a unified congestion control algorithm (CCA) set in servers cannot accommodate the heterogeneity well which degrades the quality of user experience. Especially in two scenarios: 1) Two flows in one server go through different path with completely different characteristics; 2) The user clients change their places with completely different characteristics of paths. According to the data obtained by Pantheon [4] for more than one year, the transmission performance changes greatly due to the changes in network path, bottleneck bandwidth and network delay. Whether the handcrafted CCAs or learning-based CCAs, either of these cannot be replaced by the other entirely. Some rationales of handcrafted CCAs are hard to learn by machine due to the simplification and assumption of real world. For instance, the Sprout [3] published in 2013 aims to reach high-throughput and low-latency on cellular networks. A more universal algorithm, PCC Vivace [1] proposed in 2018 does not perform better than Sprout on cellular networks. It's a good way to take the advantage of each CCA as much as possible instead of designing a comprehensive CCA to excel in most scenarios.

No matter which CCA we choose, there are two things to be done before we deploy it. At the first, We may need to make modifications to the source code of CCAs due to the different Linux kernel versions. Besides, it's extremely inconvenient if each server has to be operated respectively. Then, we should make sure that the selected CCA can get the expected performance gain in the real network which may be dynamic and hardly evaluated before.

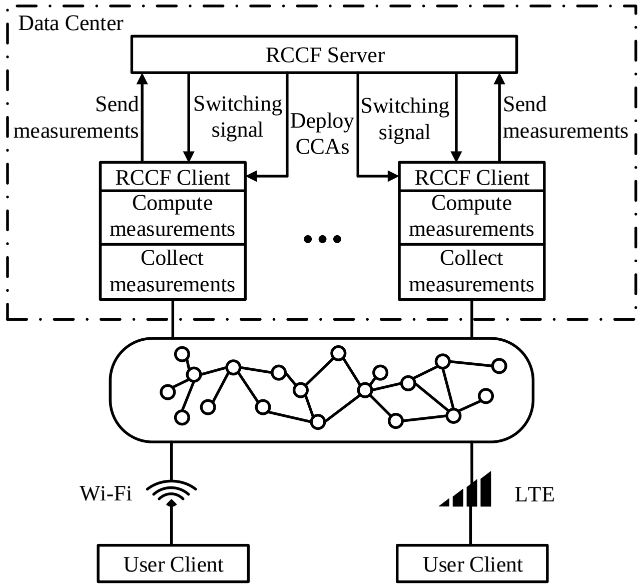

Figure 1: An overview of the structure and related functions of RCCF.

To address the above concerns, we introduce a novel congestion control framework, *Reconstructed Congestion Control Framework (RCCF)* based on CCP [2] which contains following functions: 1) Collecting run-time feedback data from various networks; 2) Switching CCAs for one flow according to diverse network environments; 3) Evaluating CCAs in a wide range of real network environments; 4) Deploying CCAs for servers easily and quickly.

2 DESIGN PRINCIPLES

An overview of the structure and related functions of RCCF is shown in Figure 1. RCCF clients are traditional servers in datacenter which send data to user clients. The RCCF server is an additionally configured server which is responsible for communicating with RCCF clients for related functions. We would like to firstly deploy RCCF in a datacenter for three reasons: 1) The CCAs of sending ends mainly determine the transmission quality and a considerable Internet traffic is sent from servers in datacenter. 2) Servers in datacenter are clustered in a centralized location which make it easy to deploy RCCF. 3) The amount of measurements transmitted by RCCF clients in datacenter is appropriate and manageable. The design principles of RCCF involve four aspects as described below.

Network measurements. Network measurements reported by one flow or one server can't cover the wide range of paths' characteristics. Some measurements collected by different servers can also be duplicated. In order to make full use of network measurements, RCCF is designed to transmit network measurements from all RCCF clients to one RCCF server. Every ACK or every RTT, RCCF clients can obtain network feedback data. For the reason that the transient measurements are useless and the frequent reports to the RCCF server are wasteful, RCCF partitions the logic of the measurements computation. A small part of computations is allocated to RCCF clients and most of them remain in the RCCF server which also evaluates the performance of CCAs through measurements.

Assign a switching task. As all network measurements are transmitted to the RCCF server, the RCCF server trains a neural network model that maps measurements to CCAs by volume data. Every time receiving the network measurements from RCCF clients, the RCCF server can select a most suitable or should evaluate CCA and assign a switching task to RCCF clients.

Deploy CCAs. The RCCF server also takes the responsibility of deploying updated and brand-new CCAs. When you finish coding the source code of one CCA and commit it to the RCCF server, the RCCF server will compile it once and release the binary callable CCA to all RCCF clients whenever it's going to be used. It has a hot update feature which is reflected in two aspects: 1) When a RCCF client is running a CCA, a modified CCA of the same name can be updated to the RCCF client and can be used immediately without the last one is done; 2) A brand-new CCA can be deployed to RCCF clients which don't know about it before and can be used directly.

Smooth switch of CCAs. The performance loss caused by transition between CCAs should be minimized. As different CCAs maintain individualized variables, we can only get part of them which can be inherited from the last CCA like RTT, threshold and so on. For all variables used by CCAs, only Congestion Window(CWND) and pacing rate eventually determine their performance. RCCF uses them when switching CCAs. If the last CCA overestimates the bottle rate or the packets we can sent, it is unfair to evaluate a newly switched CCA in this condition which may influence its further action. In addition, the minimum RTT which represents the propagation delay of path also cannot be measured accurately. Aiming to solve the above problem, RCCF sets the CWND to 10 packets for a minimum RTT to decrease the extra queueing delay at the beginning of switching. After above operations, we put the switched CCA in the bandwidth probing phase.

3 PRELIMINARY RESULTS

3.1 Ability of choosing CCAs

We test the ability of choosing the most suitable CCA by using collected network measurements including averaged throughput, averaged RTT, minimum RTT, acked packets, sacked packets, losted packets and the number of timeout. We do 1000 sets of experiments with bandwidth varies from 5Mbit/s to 50Mbit/s, propagation delay varies from 20ms to 200ms and packet loss rate varies from 0% to 10%. In each experiment, we run a single flow and switch four CCAs including TCP Reno, TCP Cubic, TCP Vegas and BBR one by one. Each CCA runs for 8s and the minimum RTT is collected during the 8s. Other network measurements are collected after 4s when the switched CCA is not affected by the last CCA.

In each experiment, we can get four sets of network measurements collected under four CCAs. We calculate the performance score of each CCA, select the CCA of highest score, and label the four sets of data with it which means the most suitable CCA with these measurements. Repeating the above operations for each set of data, we can get 4000 sets of categorical data. We use 3200 sets of data to train a simple deep neural network, and 800 sets of data to test. The 62% accuracy of the test set represents that RCCF has the ability to select the most suitable CCA through those mentioned network measurements but not enough. The accuracy should be improved by getting more elaborate measurements and designing a more effective neural network model in the future.

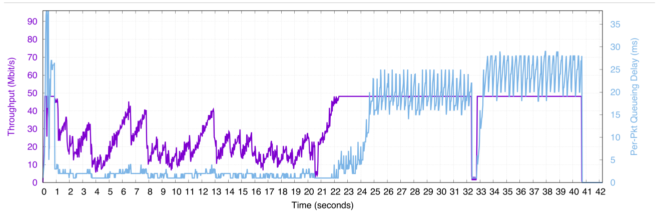

Figure 2: Switching from TCP Reno to BBR in the 0.1% packet loss rate network.

3.2 Performance Gain

We run the experiment over a 0.1% packet loss rate link with 48Mbit/s bandwidth and 20ms round trip propagation delay. It can be seen from Figure 2 that the throughput of TCP Reno is low in the first 20s. At about 21s, the RCCF client transmits network measurements to the RCCF server. After receiving the signal of switching BBR, BBR probes larger bandwidth in a short time and maintains high bandwidth utilization. By switching from TCP Reno to BBR, the bandwidth utilization of the link has more than doubled which reflects the effectiveness of switch system in RCCF.

4 CONCLUSION

We present RCCF to quickly deploy CCAs and switch CCAs for a single flow. RCCF is not only a framework for selecting CCAs and it is more capable of evaluating the performance of CCAs in real networks. Through RCCF, each CCA can make the utmost of its own advantages. Switching between CCAs and eventually achieving a greater performance over the heterogeneous and dynamic networks is our goal.

REFERENCES

[1] Mo Dong, Tong Meng, Doron Zarchy, Engin Arslan, Yossi Gilad, Brighten Godfrey, and Michael Schapira. 2018. PCC Vivace: Online-Learning Congestion Control. In *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*. Renton, WA, 343–356.

[2] Akshay Narayan, Frank Cangialosi, Deepti Raghavan, Prateesh Goyal, Srinivas Narayana, Radhika Mittal, Mohammad Alizadeh, and Hari Balakrishnan. 2018. Restructuring Endpoint Congestion Control. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication (SIGCOMM '18)*. 30–43.

[3] Keith Winstein, Anirudh Sivaraman, and Hari Balakrishnan. 2013. Stochastic Forecasts Achieve High Throughput and Low Delay over Cellular Networks. In *Presented as part of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13)*. Lombard, IL, 459–471.

[4] Francis Y. Yan, Jestin Ma, Greg D. Hill, Deepti Raghavan, Riad S. Wahby, Philip Levis, and Keith Winstein. 2018. Pantheon: the training ground for Internet congestion-control research. In *2018 USENIX Annual Technical Conference (USENIX ATC 18)*. Boston, MA, 731–743.