COC: Hierarchical Coflow Ordering for WAN Bandwidth Optimization in Inter-Data Center

Jingxuan Zhang
Tongji/Yale University

Y. Richard Yang
Yale University

CCS CONCEPTS

• **Networks → Network management**.

KEYWORDS

Coflow, Software-Defined WAN, Bandwidth Allocation

ACM Reference Format:
Jingxuan Zhang and Y. Richard Yang. 2020. COC: Hierarchical Coflow Ordering for WAN Bandwidth Optimization in Inter-Data Center . In *ACM Special Interest Group on Data Communication (SIGCOMM '20 Demos and Posters), August 10–14, 2020, Virtual Event, USA.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3405837.3411400

1 INTRODUCTION

In recent years, more and more applications involving large-scale data analytics tend to be geo-distributed [6]. Those applications distribute intermediate data into different data centers, and generate traffic going through WAN links between data centers, which are expensive resources for service providers. Many global service providers like Microsoft, Google, and Facebook have already developed their SD-WAN traffic engineering systems [5, 7, 8] to optimize the bandwidth allocation across their inter-data centers.

While existing SD-WAN systems can optimize WAN bandwidth for overall resource utilization and fairness very well, there is still a huge gap between the global bandwidth allocation and the performance of applications. As existing SD-WAN systems only require tenants to report their total traffic demand, to calculate the bandwidth enforcement strategies, SD-WAN systems treat all the flows reported by users as independent flows. However, the traffic of most applications running in modern data centers fits in the coflow model [3]. For each tenant, the per-flow rate allocation calculated by existing SD-WAN systems may not optimize its expected performance objective (*e.g.*, average coflow completion time).

To enable the SD-WAN controller to optimize performance of tenant-level coflow applications, there are three challenges:

Coordination of flow-level schedule. It is hard for the SD-WAN controller to coordinate the flow-level schedule of each tenant, as the SD-WAN can only control the tenant-level traffic on the end hosts. The flow-level traffic are controlled by tenants themselves.

Heterogeneity of coflow schedulers. Adjusting with all the complexity, the tenant may adopt different flow scheduling algorithms. Thus, it is hard for the SD-WAN controller to estimate the performance of tenants' applications.

Oscillation of rate control. Most of popular coflow schedulers [1, 2] control ordering rather than rates of coflows. The flow-level rate limits may oscillate frequently. If the SD-WAN controller still use rate control for bandwidth allocation, the oscillation will lead to scalability issue for centralized coordination.

To address those challenges, we presents COC to allow each tenant to optimize their coflow traffic as much as possible, and still guarantee the high bandwidth utilization and fairness on the WAN links of inter-data centers. COC uses a *hierachical coflow ordering architecture* to allocate WAN bandwidth for tenants. In particular, COC uses three technologies to achieve this: 1) the local tenant-level coflow coordinator that allows each tenant to schedule its own coflows, and report both coflow demands and coflow orderings, 2) the coflow ordering composition framework to coordinate tenant-level ordering, and 3) the virtual transport layer to allow the SD-WAN controller to coordinate data transfers of tenants.

In this poster, we demonstrated the overall design and the basic benefit of this architecture. We also identified potential issues to be addressed in future work.

2 SYSTEM DESIGN

We design COC for multi-tenant inter-data centers to optimize both the tenant-level coflow application performance and the WAN bandwidth allocation. As shown in Fig. 1, the architecture of COC includes a tenant-level coflow coordinator for each tenant, a tenant-level coflow daemon at each virtual host, a global coflow coordinator, and a coflow composer daemon at each physical hosts.

2.1 Hierarchical Coflow Ordering Workflow

Overall, COC applies coflow ordering in the inter-data center hierarchically. The basic workflow of COC is as follows.

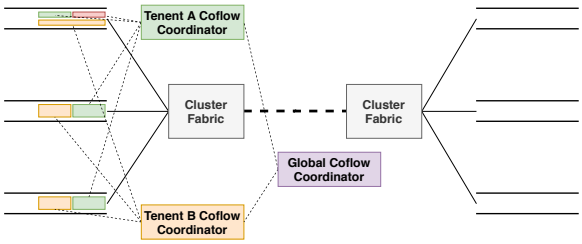
For each tenant of the inter-data center, it will logically consider the inter-data center as a single-tenant fabric and run its own coflow scheduling algorithm to determine the coflow ordering at each virtual output queue. The tenant-level coflow daemon at each virtual end host will apply all the virtual output queues to a virtual transport layer and set the priority for each flow.

At each epoch, the global coflow coordinator will read all the tenant-level coflow coordinators to get the information of the coflow demands of each tenant and the coflow ordering determined by each tenant. By running the coflow ordering composition algorithm, the global coflow coordinator computes the tenant ordering and maintains the tenant ordering list at each coflow composer daemon running on each physical host.

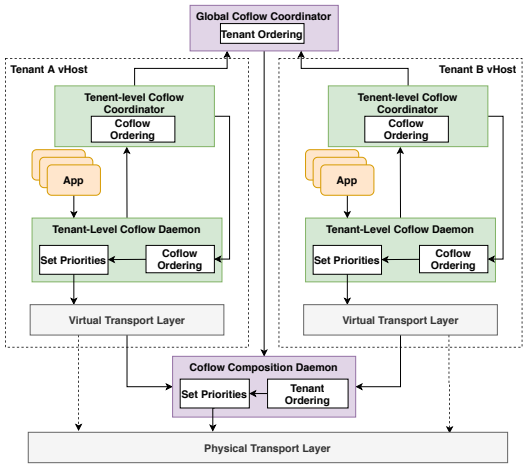
The virtual transport layer at each virtual end host will not actually execute flows, but send the determined priority to the coflow composer daemon running on its physical host. The coflow

SIGCOMM '20 Demos and Posters, August 10–14, 2020, Virtual Event, USA

ACM ISBN 978-1-4503-8048-5/20/08.
https://doi.org/10.1145/3405837.3411400



(a) Hierarchical coflow ordering architecture.



(b) Workflow of COC.

Figure 1: The architecture and workflow of COC.

composer daemon will reassign the flow priorities based on the tenant ordering maintained by the global coflow coordinator, and send flows to the transport layer of the physical host.

2.2 Tenant-level Coflow Ordering

The components of tenant-level coflow ordering in COC references the design of Sincronia [1]. We reuse the implementation of the Sincronia daemon as the tenant-level coflow daemon, and extend the central coordinator of Sincronia as the tenant-level coflow coordinator to support APIs to interactive with the global coflow coordinator.

2.3 Coflow Ordering Composition

The main component of COC is the global coflow coordinator. Its main function is to compute the coflow ordering composition. To perform the coflow ordering composition, the global coflow coordinator divides the time into epochs. At each epoch, the global coflow coordinator reads the coflow information registered to each tenant and the "ordered flows" list from each tenant-level coflow coordinator. Then it models the coflow ordering of each tenant as the virtual output queue from each ingress port to each egress port. For example, the left side of Fig. 2 shows the virtual output queues of two tenants.

Then the global coflow coordinator uses a greedy algorithm – Maximum-Welfare-Tenant-First (MWTF) – for composing the virtual output queues of multiple tenants. The welfare of a tenant is defined as the estimated Coflow Completion Time (CCT) of the tenant multiplying the number of coflows of the tenant. Fig. 2

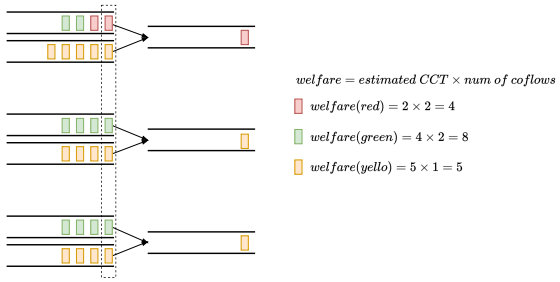


Figure 2: Coflow ordering composition using the Maximum-Welfare-Tenant-First (MWTF) algorithm.

shows the example of coflow ordering composition using the MWTF algorithm.

2.4 Virtual Transport Layer

The execution of the composed coflow ordering relies on the virtual transport layer running on the virtual hosts. The virtual transport layer of COC does not really execute flows for the tenant, but just maintains the meta data (tenant-level priority, coflow id, target egress port, etc.) of each tenant flow, and provides callback functions to start the flow transmission. At each epoch, based on the tenant ordering list maintained by the global coflow coordinator, the coflow composition daemon determines which tenant will send flows to each egress port. For the determined tenant, the coflow composition daemon will find out the current coflow expected to be sent by this tenant, and invoke the callback function of the virtual transport layer of the tenant to start the coflow.

3 PRELIMINARY RESULTS

We simulated COC on a simple inter-data center setting with two data center fabrics connected via a single WAN link. We synthesized 4000 coflow traces based on the Facebook workload used by [4] and randomly separated coflows into three tenants. We evaluated the MWTF algorithm with those traces to compare with the MPFA algorithm used by BwE [7]. Using COC with the MWTF algorithm, the coflow completion time of each tenant is reduced by 20% on average. And the bandwidth utilization of the WAN link is 100%.

4 DISCUSSION AND FUTURE WORK

This poster proposed a novel architecture to coordinate the SD-WAN controller and the coflow applications of tenants. The preliminary results demonstrated its benefits. However, the scalability with increasing numbers of tenants and (co)flows have not been studied yet. And because of the resource competition among tenants, how to guarantee strategy-proof should also be studied. As our future work, we will improve our system from these aspects.

ACKNOWLEDGMENTS

The research of Y. Richard Yang is supported in part by a Facebook Networking Systems award and the project "PCL Future Greater-Bay Area Network Facilities for Large-scale Experiments and Applications (LZC0019)".

REFERENCES

[1] Saksham Agarwal, Shijin Rajakrishnan, Akshay Narayan, Rachit Agarwal, David Shmoys, and Amin Vahdat. 2018. Sincronia: near-optimal network design for coflows. In *Proceedings of SIGCOMM '18*. ACM Press, New York, New York, USA, 16–29. https://doi.org/10.1145/3230543.3230569

[2] Mohammad Alizadeh, Shuang Yang, Milad Sharif, Sachin Katti, Nick McKeown, Balaji Prabhakar, and Scott Shenker. 2013. PFabric: Minimal near-Optimal Datacenter Transport. *SIGCOMM Comput. Commun. Rev.* 43, 4 (Aug. 2013), 435–446. https://doi.org/10.1145/2534169.2486031

[3] Mosharaf Chowdhury and Ion Stoica. 2012. Coflow: a networking abstraction for cluster applications. In *Proceedings of HotNets-XI*. ACM Press, New York, New York, USA, 31–36. https://doi.org/10.1145/2390231.2390237

[4] Mosharaf Chowdhury, Yuan Zhong, and Ion Stoica. 2014. Efficient Coflow Scheduling with Varys. *SIGCOMM Comput. Commun. Rev.* 44, 4 (Aug. 2014), 443–454. https://doi.org/10.1145/2740070.2626315

[5] Chi-Yao Hong and et al. 2013. Achieving high utilization with software-driven WAN. In *Proceedings of SIGCOMM '13*. ACM Press, New York, New York, USA, 15. https://doi.org/10.1145/2486001.2486012

[6] Chi-Yao Hong and et al. 2018. B4 and after: Managing Hierarchy, Partitioning, and Asymmetry for Availability and Scale in Google's Software-Defined WAN. In *Proceedings of SIGCOMM '18 (SIGCOMM '18)*. ACM Press, New York, NY, USA, 74–87. https://doi.org/10.1145/3230543.3230545

[7] Alok Kumar and et al. 2015. BwE: Flexible, Hierarchical Bandwidth Allocation for WAN Distributed Computing. In *Proceedings of SIGCOMM '15*, Vol. 45. ACM Press, New York, New York, USA, 1–14. https://doi.org/10.1145/2785956.2787478

[8] Brandon Schlinker and et al. 2017. Engineering Egress with Edge Fabric: Steering Oceans of Content to the World. In *Proceedings of SIGCOMM '17*. ACM Press, New York, New York, USA, 418–431. https://doi.org/10.1145/3098822.3098853