UCAT: User Centric Adaptive Transmission for Meeting Diverse Network Demands

Hanxiao Yan[1], Chengxiao Yu[2(✉)], Kang Liu[2], Deyun Gao[1], Du Chen[1], and Haoran Song[1]

[1] School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing 100044, China
{hanxiaoyan,19111028,gaody,haoransong}@bjtu.edu.cn
[2] Department of New Networks in Peng Cheng Laboratory, Shenzhen, China
yuchx@pcl.ac.cn

Abstract. With the development of network, many emerging network applications such as telemedicine and Virtual Reality (VR) have put forward higher requirements on network bandwidth. However, the conventional coarse-grained service model of networks faces challenges in meeting the diverse and dynamic bandwidth requirements posed by these applications. For instance, while simple text browsing demands minimal bandwidth, bandwidth-intensive applications like VR require substantially higher resources. In this paper, we propose UCAT, an efficient multipath transmission architecture for user bandwidth demand adaptation. Specifically, UCAT leverages in-band network telemetry (INT) to collect network status information and generate multiple forwarding paths, which splits user traffic into flowlets and performs weighted scheduling. By customizing the packet transmission format using P4, UCAT achieves bandwidth demand awareness, enabling users with a single network interface card to achieve higher transmission bandwidth. Extensive experimental results show that UCAT can effectively meet user bandwidth requirements, reduce disordered packets, and improve network throughput.

Keywords: Concurrent multipath transfer · Flowlets · INT · P4

1 Introduction

With the rapid evolution of the Internet, a multitude of diverse applications have emerged, each placing higher demands on network bandwidth. However, traditional networks that rely on a single path for user traffic struggle to meet these varied bandwidth requirements. To address this challenge, multipath transmission has emerged as a promising solution, enabling users to simultaneously send their traffic across multiple paths. Multipath transmission offers numerous advantages, including load balancing, increased network throughput, and improved reliability [1].

V. C. M. Leung et al. (Eds.): Qshine 2023, LNICST 574, pp. 15–25, 2024.
https://doi.org/10.1007/978-3-031-65123-6_2

In the realm of multipath transmission, multipath TCP (MPTCP) has been widely adopted for packet-level traffic scheduling. However, the asymmetric nature of heterogeneous paths often leads to a significant number of out-of-order packets, potentially congesting the receive buffer and degrading the performance of MPTCP transmission [2]. Moreover, implementing MPTCP typically requires multiple network interface cards (NICs) and incurs substantial costs for complex traffic scheduling at the endpoint. Recently, several studies have explored flowlet-based scheduling to mitigate packet disorder. But these approaches lack accurate and real-time network status information, limiting their ability to further enhance bandwidth performance. Leveraging software-defined networking (SDN) and programmable data planes, in-band network telemetry (INT) [3] has emerged as an efficient means of achieving precise and real-time network monitoring.

In this paper, we propose an efficient multipath transmission system based on INT, which dynamically adapts to user bandwidth demands. Our system selects one or more paths based on network conditions to fulfill user bandwidth requirements, employing flowlet-based scheduling for user flows. The key contributions of this paper are summarized as follows.

- We design a concurrent multi-path transfer system with user intent adaptation. The system resolves user bandwidth requirements and makes best efforts to guarantee them by scheduling user flows over multiple paths.
- We propose a transport mechanism that combines INT with flowlet scheduling. We use INT to collect network-wide status information based on our previous work to provide the necessary data support for system path planning. To reduce packet disorder, we adopt flowlet as the scheduling unit for flows on multiple paths.
- We implement the prototype of the system on BMv2 with the P4Runtime. Detailed experimental results prove the outstanding performance of the system in terms of user bandwidth adaptation and concurrent multi-path transmission.

The remainder of this paper is organized as follows. Section 2 elaborates the related work. Section 3 introduces the design of system in details, while the experimental environment and results are presented in Sect. 4. Finally, Sect. 5 concludes this paper.

2 Related Work

In this section, we provide brief introduction for the technologies that are leveraged in this work.

2.1 In-Band Network Telemetry

As network traffic continues to grow in volume and complexity, traditional network measurement methods struggle to meet the demands of efficient network

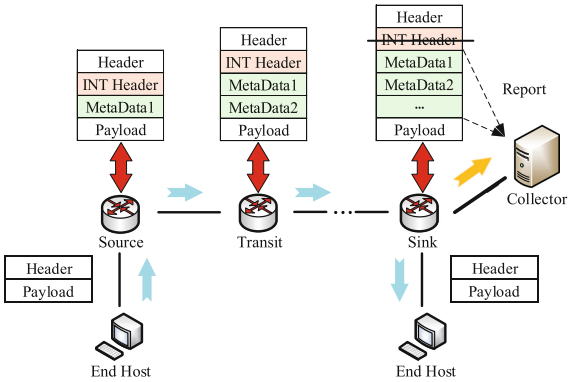

Fig. 1. The Original INT Architecture.

management. These methods rely on out-of-band data collection methods that require dedicated monitoring devices and can introduce latency and overhead into the network. With the advent of Software Defined Network (SDN) and programmable data plane, INT is proposed as a promising solution. Figure 1 illustrates how, unlike traditional network measurement, INT integrates packet forwarding and network measurement to provide real-time and accurate network visibility without requiring additional equipment [4].

However, because Original INT relies heavily on business packets, it is almost impossible to achieve satisfactory in-band network-wide telemetry. In recent years, there have also been some researches trying to solve this problem, such as INT path [5], INT filter [6] and so on. In order to provide fine-grained network-wide traffic monitoring, we also proposed cluster-based fast in-band network-wide telemetry (CFINT) [7] as an improvement of the original INT framework, which can overcome the MTU limitation and significantly reduce the network-wide telemetry latency.

2.2 Multipath Routing and Flowlet

Traffic splitting algorithms for multipath routing can be broadly classified into three types: packet-level, flow-level, and flowlet-level [8]. As a typical packet-level application, multipath TCP (MPTCP) is often affected by heterogeneous links that cause packet congestion, which can degrade transmission performance by blocking the receive buffer [2]. Flow-based solutions such as WCMP [9] confine a flow to one path without the risk of congestion. However, it is not suitable for concurrent multipath transmission between a source-destination pair, and it is difficult to guarantee load balancing in the case of a small number of flows. Flowlets are bursts of packets from a flow separated by sufficiently large gaps. This avoids the problem of out-of-order packets while still providing reasonably fair traffic distribution [10]. Between the packet level and the flow level, flowlet seems to be a better solution for concurrent multipath transmission and load balancing.

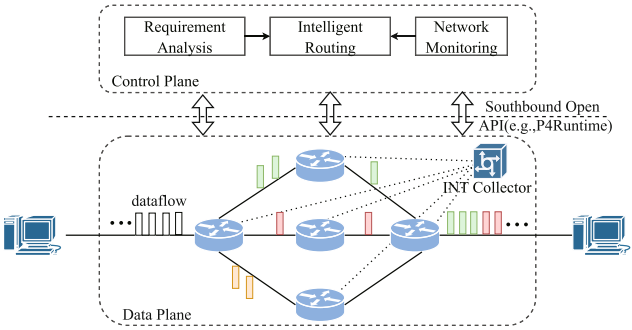

Fig. 2. The System Architecture.

Shi J et al. [11] proposed a dynamic multipath forwarding mechanism with flowlet granularity for the heterogeneous Internet of Things. They extend the programmable data plane to maintain state information to implement congestion-aware networks and flowlet-based dynamic forwarding.

F. Fan et al. [12] propose the Dynamic Roulette Wheel (DRW), a load-aware balancing algorithm for multipath data center networks. They dynamically divide flows into flowlets based on link load to improve performance under non-uniform and dynamic traffic. They also use roulette wheel selection to determine a forwarding path to improve bandwidth utilization and address the bottleneck problem.

3 Design of System

To satisfy the user's intention, we provide a network bandwidth customized transmission system to make the network more adaptable to the user. The overall system architecture is shown in Fig. 2. The network controller analyzes the demand header and obtains the user's bandwidth demand. The intelligent routing module generates a set of paths that attempt to meet the demand based on the obtained demand bandwidth and network status information, and sends them to the forwarding plane. The data plane switch splits the user data stream into flowlets and forwards them in multiple paths. In addition, the network monitoring module collects network status information in the form of int all the time. Next, we will discuss the implementation of each module.

3.1 Network Monitoring Module

The network monitoring module provides network status data to support route planning. Based on the previous work, we use CFINT to ensure fine-grained and timely collection of network-wide status information.

First, CFINT divides the network topology into multiple clusters using minimum dominating set. Then, according to the fact that the forwarding path from

cluster head to cluster head will not exceed four hops, CFINT generates a series of short forwarding paths with similar lengths. In addition, CFINT uses source routing to forward the telemetry probe on demand. Required switch internal states should be inserted into the probe, and the probe is sent to the collector at the sink switch. Extensive experimental results show that CFINT can achieve synchronous multipath telemetry and outperforms existing solutions in terms of network-wide telemetry latency.

We define the link bandwidth as the number of bits transmitted per second. The average current bandwidth can be calculated from the Formula 1. tx_t is the amount of bits transmitted by the link at time t. b_{used} is the ratio of the difference between the bits transmitted at moments t_2 and t_1 to the difference between t_2 and t_1. The available bandwidth of link b_a can be determined by the Formula 2 as follows where $b_{capacity}$ represents the capacity of the link:

$$b_{used} = \frac{tx_{t_2} - tx_{t_1}}{t_2 - t_1} \tag{1}$$

$$\mathrm{b}_a = b_{capacity} - b_{used} \tag{2}$$

We collect multidimensional information including node-id at the device level as well as egress port timestamp and packet length at the port level by CFINT. Furthermore, we can calculate the value of b_{used} and b_a. With CFINT's periodic probe packets, we can easily get a snapshot of the available bandwidth on the entire network link and other status parameters.

3.2 Intelligent Routing Module

Users add demand headers to the packets in the flow describing the bandwidth they need. After obtaining and analyzing the user demand parameters, we propose a path generation algorithm that tries to find one or more paths to meet the user's bandwidth requirement. Since we focus on meeting user bandwidth requirement, disjoint paths are not necessary and the existence of overlapping links in the paths is tolerable. The pseudo-code of the path generation algorithm is shown in Algorithm 1. The algorithm is a multi-restart heuristic based on simulated annealing that aims to find a set of paths that satisfy the user's needs as much as possible, with the minimum possible maximum delay difference between paths and the fewest possible number of hops.

Firstly, We use the $\epsilon - greedy$ strategy in the single path generation process to choose between a random path or a path generated by the A^* algorithm. If the path satisfies the delay constraints then candidate solutions are added. Finally the optimal solution is taken by simulated annealing approach. The cost for path selection is set as shown in Formula 3, where *scaled_bw* and *scaled_delay* are the bandwidth and delay after the min-max normalization. For a path consisting of n links (l_1, l_2,..., l_n), the available bandwidth of the path can be calculated by Formula 4. To ensure transmission of other small streams, b_{a_n} is set to a discounted value of the actual available bandwidth. To refine the evaluation of a solution, we set the energy function in simulated annealing as shown in Formula

Algorithm 1: Bandwidth-Delay Constrained Path Searching algorithm

Input: network topology G, source node s, destination node d, desired bandwidth B_d, link available bandwidth $G(link, b_a)$, link delay $G(link, d)$
Output: path set P_s

Initialization: initial temperature $T = T_0$, cooling rate $r = r_0$, exploration times $epoch = epoch_0$, running times $run = run_0$, explore probability $\epsilon = \epsilon_0$, maximum delay difference $\Delta t_{max} = \Delta t_{0max}$, maximum number of paths $pn_{max} = pn_{0max}$, Reheating rate $h = h_0$
for *1 to* run_0 **do**
    best path list $P_{bl} = []$
    current path list $P_{cl} = []$
    for i *= 1 to* $epoch_0$ **do**
        candidate path list $P_{ca} = []$
        while *current bandwidth* $B_c < B_d$ *and current number of paths* $pn_c < pn_{0max}$ **do**
            if $random(0, 1) < \epsilon_0$ **then**
                p = random_path_selection(G,s,d)
            end
            else
                update the weights of the edges in the graph
                p = astar_path(G,s,d)
            end
            P_{ca}.append(p)
            if $P_{ca}.maxDelayRange < \Delta t_{0max}$ **then**
                bw = get_path_bandwidth(p)
                update_graph_state(G,p,bw)
            end
            else
                P_{ca}.remove(p)
            end
        end
        $E = get\_energy(P_{cl})$
        $E^* = get\_energy(P_{ca})$
        $\Delta E = E^* - E$
        $p_{\text{accept}} = e^{-\Delta E/T_0)}$
        if $p_{accept} > rand(0, 1)$ *or* $p_{accept} = inf$ **then**
            $P_{cl} = P_{ca}$
        end
        if $get\_energy(P_{cl}) < get\_energy(P_{bl})$ **then**
            $P_{bl} = P_{cl}$
        end
        if $i \bmod n == 0$ **then**
            $T_0 = T_0 * h_0$
        end
        $T_0 = T_0 * r_0$
    end
    if $get\_energy(P_{bl}) < get\_energy(P_s)$ **then**
        $P_s = P_{bl}$
    end
end
return P_s

5. This formula combines bandwidth, delay, number of paths and number of hops.

$$\text{cost} = \alpha_1 * \text{scaled\_delay} + \alpha_2 * (1 - \text{scaled\_bw}) \tag{3}$$

$$\text{bw} = \min(b_{a_1}, b_{a_2}, ..., b_{a_n}) \tag{4}$$

$$energy = \begin{cases} \alpha_1 * scaled\_delay\_range + \alpha_2 * path\_num + \alpha_3 * hop \\ \qquad \text{if } path\_num > 1 \text{ and } B_c > B_d \\ \beta_1 * scaled\_daley\_range - \beta_2 * scaled\_bw + \beta_3 * hop + \beta_4 * path\_num \\ \qquad \text{if } path\_num > 1 \text{ and } B_c \leq B_d \\ \gamma_1 * scaled\_bw + \gamma_2 * hop \\ \qquad \text{if } path\_num = 1 \end{cases} \tag{5}$$

For efficient transmission, we use a weighted approach to distribute traffic among multipaths. The path weight is calculated by Formula 6. In the formula, bw_i represents the bandwidth allocated to $path_i$ which is available in path bandwidth set B_p, w_i represents the weight of the $path_i$.

$$\text{w}_i = \frac{bw_i}{\sum_{j=1}^{n} bw_j} \tag{6}$$

The results, consisting of a weight set W_s and a path set P_s, are passed to the data plane by P4Runtime to guide forwarding.

3.3 Flowlet-Based Forwarding Module

The data plane uses regular table forwarding for user traffic that does not have a demand header. For traffic with this header, the data plane first groups requests using quadruples (source IP address, destination IP address, source port, destination port) and then maps the set of paths passed by the controller into tunnels, after which the same group of traffic is switched between multiple tunnels and forwarded as flowlets.

As for the path weights, we use the tunnel table entry range matching implementation to reduce the waste of table space in the WCMP [9] method. Cumulative weights are used as bounds for range matching. When a new flowlet arrives, the hash value generated from the packet header fields (source address, destination address, source port, destination port, flowlet id) is compared to the bounds and determines the tunnel it matches and the associated path.

4 Performance Evaluation

4.1 Overview

UCAT is implemented in mininet using BMv2 with P4Runtime. In order to evaluate the performance of the system, we use the experimental topology shown in Fig. 3 and use iperf as the bandwidth measurement tool.

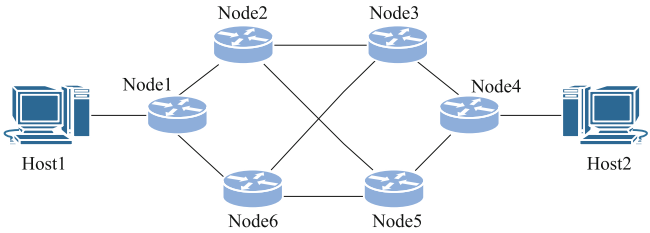

Fig. 3. The Experimental Topology.

4.2 Simulation Results

First, we compare the Bandwidth-Delay Constrained Path Search-ing algorithm, the DFS algorithm used by Chahlaoui et al. [13], and the path planning algorithm used by B.Yan et al. [14]. The path generation results are shown in Table 1. The algorithms of Chahlaoui et al. and B. Yan et al. only support the specification of the number of paths as a parameter input. To compare their results with ours, we enter the same number of paths in their algorithms. However, their algorithms use static path planning without considering network state information. As a result, they are unable to dynamically plan paths according to different requirements. In addition, the efficiency of multipath transmission is closely related to the state information between paths. The paths generated by their algorithms are more differentiated, which can lead to packet disorder during TCP transmission. In contrast, our algorithm generates a set of optimal paths that minimize the maximum delay difference between paths and reduce the differences between paths while ensuring that user requirements are met as much as possible. Therefore, it achieves more favorable transmission performance in the actual transmission.

Table 1. Comparing The Path Generation Algorithms

Required Bandwidth (Mbps)	Planned Bandwidth (Mbps)			Max Latency Difference (ms)		
	UCAT	DFS	PPA	UCAT	DFS	PPA
5	8	10	10	0	0	0
10	9.6	15	20	0	15	7
15	12	15	20	2	15	7
20	16	15	20	7	15	7
25	24	15	20	2	15	7

For a single 10 Mbps transmission bandwidth requirement, simulation tests were performed and compared with a single transmission based on maximum bandwidth prioritization and a Round-Robin (RR) based transmission scheme. The results obtained are shown in Fig. 4 and Fig. 5. For UDP, a protocol that

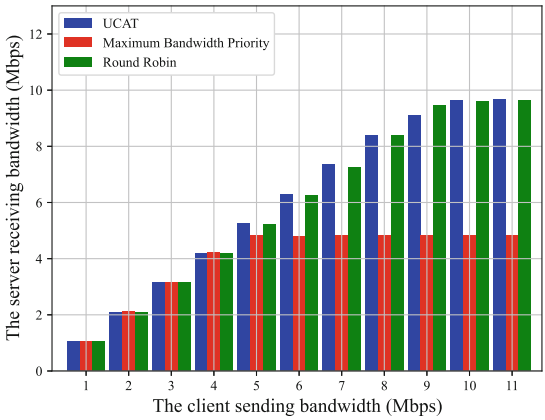

Fig. 4. Receive Bandwidth of UDP.

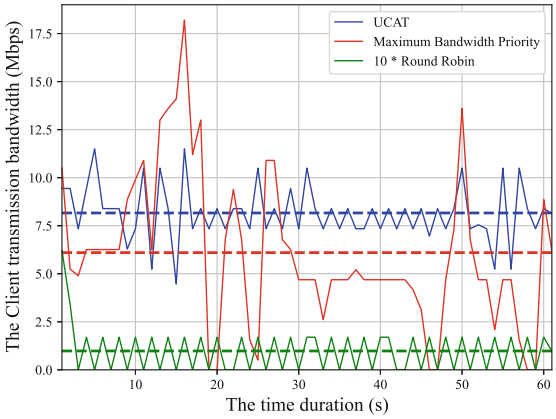

Fig. 5. Send Bandwidth of TCP Client.

does not take packet disorder into account, it can be seen that RR and UCAT transmit similarly, and are far better than maximum bandwidth-first single-pass transmission under high traffic. However, in TCP transmission, RR's transmission efficiency deteriorates dramatically. This is because that RR is the simplest scheduling algorithm, which sends RR can achieve a good performance in the scenario where the paths have homogeneous transmission characteristics. However, the paths are heterogeneous in most of the real-network scenarios. Using RR will cause a wealth of out-of-order packets. And the flowlet based can avoid disorder by utilizing the microburst property in a TCP transmission. Figure 6 shows the change in bandwidth usage of a single path over time when UCAT transmits over multiple paths, and it can be clearly noticed that UCAT equalizes the large bandwidth traffic to multiple paths, resulting in a more balanced network load.

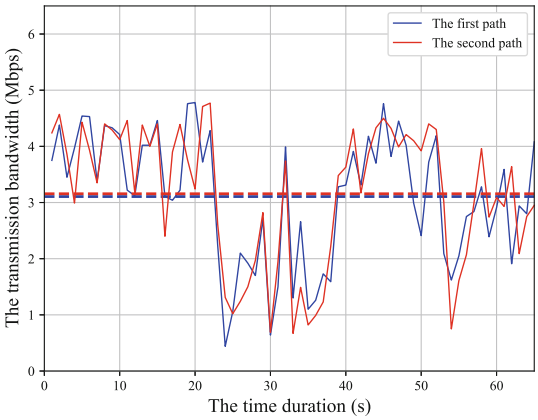

Fig. 6. Single Path Bandwidth Variation.

5 Conclusion

In this paper, we propose UCAT to generate optimal multipath paths for different user bandwidth requirements. With the accurate and real-time network status information provided by INT, UCAT first uses simulated annealing heuristic algorithms to generate paths with smaller delay difference for users that satisfy their bandwidth demands as much as possible. Then, UCAT splits the user traffic into flowlets as the basic unit of concurrent transmission, which not only avoids long time occupancy of a single path, but also reduces disordered packets. Extensive experiments show that UCAT can effectively meet user bandwidth requirements while balancing the network load.

Acknowledgment. This work is supported in part by the Fundamental Research Funds for the Central Universities (Grant No. 2022JBGP002), supported in part by the National Natural Science Foundation of China (grant no. 61971028), supported in part by the Major Key Project of PCL under grant PCL2022Y04. Corresponding author: Chengxiao Yu (yuchx@pcl.ac.cn).

References

1. Hodroj, A., Ibrahim, M., Hadjadj-Aoul, Y.: A survey on video streaming in multipath and multihomed overlay networks. IEEE Access **9**, 66816–66828 (2021)
2. Morawski, M., Ignaciuk, P.: Choosing a proper control strategy for multipath transmission in industry 4.0 applications. IEEE Trans. Ind. Inf. **9**(6), 3609–3619 (2022)
3. Kim, C., Sivaraman, A., Katta, N., Bas, A., Dixit, A., Wobker, L.J.: In-band network telemetry via programmable dataplanes. In: ACM SIGCOMM 2015, London, United Kingdom, vol. 15, pp. 1–2 (2015)
4. In-band Network Telemetry (INT) Dataplane Specification. https://p4.org/p4-spec/docs/INT_v2_1.pdf. Accessed 25 June 2020

5. Pan, T., et al.: INT-path: towards optimal path planning for in-band network-wide telemetry. In: INFOCOM 2019, Paris, France, pp. 487–495 (2019). https://doi.org/10.1109/INFOCOM.2019.8737529
6. Song, E., et al.: Int-filter: mitigating data collection overhead for high-resolution in-band network telemetry. In: GLOBECOM 2020, Taipei, China, pp. 1–6 (2020). https://doi.org/10.1109/GLOBECOM42002.2020.9348029
7. Chen, D., Gao, D., Foh, C.H., Yan, H.: CFINT: cluster based fast in-band network-wide telemetry in 6g-enabled networks. In: Globecom Workshops 2022, Rio de Janeiro, Brazil, pp. 1699–1704 (2022). https://doi.org/10.1109/GCWkshps56602.2022.10008610
8. Vanini, E., Pan, R., Alizadeh, M., Taheri, P., Edsall, T.: Let it flow: resilient asymmetric load balancing with flowlet switching. In: NSDI 2017, Boston, MA, USA, pp. 407–420 (2017)
9. Zhou, J., et al.: WCMP: weighted cost multipathing for improved fairness in data centers. In: The Ninth European Conference on Computer Systems 2014, New York, NY, USA, pp. 1–14 (2014). https://doi.org/10.1145/2592798.2592803
10. Alizadeh, M., et al.: Conga: distributed congestion-aware load balancing for datacenters. In: SIGCOMM 2014, Chicago, Illinois, USA, pp. 503–514 (2014). https://doi.org/10.1145/2619239.2626316
11. Shi, J., et al.: Flowlet-based stateful multipath forwarding in heterogeneous internet of things. IEEE Access **8**, 74875–74886 (2020)
12. Fan, F., Meng, H., Hu, B., Yeung, K.L., Zhao, Z.: Roulette wheel balancing algorithm with dynamic flowlet switching for multipath datacenter networks. IEEE/ACM Trans. Network. **29**(2), 834–847 (2021)
13. Chahlaoui, F., Dahmouni, H., El Alami, H.: Multipath-routing based load-balancing in SDN networks. In: CIoT 2022, Marrakech, Morocco, pp. 180–185 (2022). https://doi.org/10.1109/CIoT53061.2022.9766801
14. Yan, B., Liu, Q., Shen, J., Liang, D.: Flowlet-level multipath routing based on graph neural network in openflow-based SDN. Future Gener. Comput. Syst. **134**, 140–153 (2022)