Toward QoE-based Routing Path Selection

Umakant Kulkarni†, Yufeng Chen†, Patrick Melampy‡, Sonia Fahmy†
†*Purdue University* ‡*Juniper Networks*
e-mail: {ukulkarn, chen4044, fahmy}@purdue.edu, pmelampy@juniper.net

***Abstract*—The increasing popularity of video streaming and conferencing services have altered the nature of Internet traffic. In this paper, we take a first step toward quantifying the impact of this changing nature of traffic on the Quality of Experience (QoE) of popular video streaming and conferencing applications. We first analyze the traffic characteristics of these applications and of backbone links, and show how simple multipath routing may adversely impact application QoE. To mitigate this problem, we propose a new routing path selection approach, inspired by the TCP timeout computation algorithm, that uses both the average and variation of path load. Preliminary results show that this approach improves application QoE by on average 14% and packet latency by 11% for video streaming and conferencing applications, respectively.**[1]

I. Introduction

Internet Service Providers (ISPs) have traditionally leveraged intra-domain routing protocol *weights* or MultiProtocol Label Switching (MPLS) for traffic engineering [1]. Traffic engineering operations between the ingress and egress points of an ISP are typically performed over relatively long time scales. Equal Cost MultiPath (ECMP) routing has been an important component of shorter time-scale traffic engineering and load balancing in both the wide-area [2], [3] and data center [4], [5], [6], [7] contexts. ECMP is attractive due to its simple implementation. The ultimate goal of this work is to augment ECMP for better traffic engineering and application QoE in the wide area.

An important development that impacts traffic engineering is that video streaming and conferencing applications now constitute a majority of traffic [8], [9]. The traffic characteristics and Quality of Experience (QoE) requirements of these applications should therefore play a major role in traffic engineering. This is because application QoE is an important indicator of customer satisfaction. For instance, the Video Multi-Method Assessment Fusion (VMAF) metric [10], proposed by Netflix, assesses perceptual video quality. In addition to traditional metrics such as link utilization, packet loss, end-to-end delay and jitter, an ISP needs to ensure that its customers have high QoE for the most popular applications.

Motivated by this observation, we strive to understand the factors that impact the QoE of video streaming and conferencing applications. In addition to the load on the links on the path that the traffic traverses, we posit that the burstiness of the background traffic on these links must also play a key role in traffic engineering.

[1]This work has been supported in part by a gift from Juniper Networks. The authors would like to thank George Cybenko (Dartmouth College) and Kieran Milne (Juniper Networks) for discussions of this work.

Significant research has been conducted on evaluating traffic burstiness [11], [12], [13], [14], [15]. "Self-similarity" of traffic at different time scales and at different levels of flow aggregation has been the subject of major debate in the 1990s [16], [17] and into the 21st century [18], [19]. Based on the importance of these burstiness measures, we design a path selection approach that considers both average load and background traffic variation on a network path when selecting among multiple, equal-cost, paths for routing traffic of video streaming or conferencing applications.

This paper makes the following contributions: (1) We examine the traffic characteristics of popular applications (Sections II) and backbone links (Section III). (2) We describe two path selection approaches, one solely based on average load and another that incorporates both the average and variation of the load (Section IV). (3) We conduct a series of experiments to compare these two approaches with basic ECMP routing (Sections V and VI), focusing on the impact of path selection and background traffic burstiness on application QoE. Section VII summarizes related work, and Section VIII discusses directions for future work.

II. Streaming and Conferencing Traffic

To understand the traffic characteristics of increasingly-popular conferencing and streaming, we measure seven conferencing services (Zoom, Cisco WebEx, Slack, Microsoft Teams, Skype, Discord, and Google Meet), and five video streaming services (Disney+, HBO Max, Hulu, Peacock, and Prime).

Our data collection process involves two users in West Lafayette, IN, USA: user-1 connects via a cellular network on T-Mobile with iPhone XS, whereas user-2 connects to a Wi-Fi network on Comcast-Xfinity using a Windows-10 desktop. The packets travel through different autonomous systems and different access networks. We analyze both control and data plane messages, and experiment with combinations of audio, video, and screen share feeds for conferencing services. We attempt to keep the audio, video, and screen share content largely similar across our experiments (of duration 15 minutes each), which we conducted in March–April 2022.

We find that conferencing services create at least three bidirectional UDP flows, regardless of enabled feeds. This ensures that if a user enables a new feed during a session, the service can start data transfer right away. Teams and WebEx always create four bidirectional flows instead of three. Google with screen sharing creates four bidirectional flows.

Analysis of streaming traffic reveals that applications, once authorized and have obtained the URL for the media, create a

TABLE I: Characteristics of conferencing traffic

Service	Mbps	Loss %	Disp.	Kurtosis	Skewness	Prom.
Discord	1.9	0.041	31.44	1288	-1.22	268
Google	1.64	0.002	11.54	2113	-3.14	117
Skype	3.36	0.09	31.21	57071	-0.68	360
Slack	1.7	-	7.69	1193	-3.29	256
Teams	4.3	0.018	40.19	5171	-2.12	458
WebEx	0.79	0.018	5.43	49	-1.09	104
Zoom	1.73	-	16.92	15076	-3.5	177

TABLE II: Characteristics of streaming traffic

Service	Mbps	Dispersion	Kurtosis	Skewness	Prominence
Disney	4.66	7000	4584	2.71	3876
HBO	8.56	3902	4610	1.39	3943
Hulu	3.02	3611	10031	2.41	1228
Peacock	8.44	9370	25375	1.79	2449
Prime	2.52	655	420	4.28	2381

single TCP flow at a time to download the video in "chunks." Differences in throughput values between Tables I and II imply that a routing path would be more highly utilized by a single large streaming flow, compared to the small media flows of conferencing applications.

Our findings confirm that application sessions comprise "mice" flows (audio and control signaling) and "elephant" flows (video). These flows vary not only in total bytes transferred and number of packets sent, but also in the time between successive packets. Therefore, we take each 5-tuple-based unidirectional flow and study the time series of the inter-packet times and the packets sent per second. Flow pictures (FlowPics) [20] such as those depicted in Fig. 1 show the packet-size distribution over time. Characteristics of such flow time series data have been studied in previous work, which computes the dispersion [11] and kurtosis [13] for the inter-packet times, and the skewness [12], [14] and prominence [15] for the packets-per-second time series.

We observe that conferencing services send UDP real-time traffic, yielding lower dispersion in their inter-packet times (Table I) than TCP-based streaming (Table II). Kurtosis measures the outliers in the inter-packet times [13]. Table I reveals that Skype traffic and Zoom traffic have the highest number of outliers among conferencing services; other traffic appears to have been paced or shaped. Conferencing services have negative skew, whereas streaming services have positive skew, since they send significant traffic at the start of a session. Prominence values denote the maximum number of packets sent in a burst. TCP behavior in streaming services results in higher prominence than conferencing services, which deliver data at approximately constant bit rate.

We posit that an ISP that considers such time series characteristics can ensure that a service can be accommodated with high QoE on a given path. For example, sending video streaming flows via a path with already multiple high-prominence flows, and hence high burstiness, can adversely impact QoE. We validate this hypothesis in Section VI.

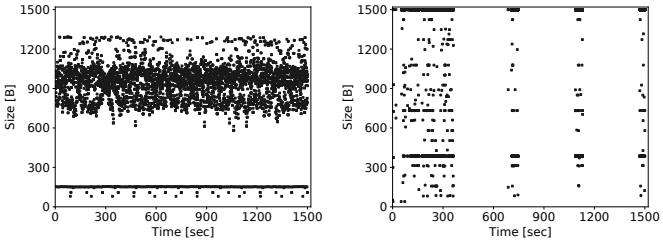

Fig. 1: Zoom (left) and Disney+ (right) flow pictures [20] show the smoothness of conferencing traffic compared to the bursty nature of streaming traffic.

III. Backbone Traffic

We now study the traffic characteristics of backbone links which include aggregates of conferencing and streaming traffic, as well as other applications. We find that many (but not all) of our results are consistent with earlier findings [16], [17], [18], [19]. We begin by visualizing the traffic traces and autocorrelation functions across different time scales. Next, we calculate the Hurst parameter to understand the intensity of long-range dependence (LRD). Finally, we show the importance of selecting an appropriate time scale by computing the same characteristics as in Section II.

We use seven publicly available traffic traces, each lasting 15 minutes. These traces are obtained from the WIDE backbone [21] and represent a day of network activity for each day of the week between March 6 and March 12, 2023.

Similar to previous research [16], [19], we explore two possible metrics for aggregating time series data: the number of bytes or the number of packets per time unit. We find that both metrics yield comparable results. Therefore, for the remainder of this section, we focus on aggregating the number of packets transferred over a designated time interval. Specifically, we count the number of packets every millisecond in the original time series.

Fig. 2 displays the number of packets sent at various time scales for the trace SF09, with each plot comprising 1500 data points. We find that traffic is not smooth even at large time scales, and exhibits burstiness across all time scales. The autocorrelation coefficients (ACFs) for the entire trace are depicted in Fig. 3 (left). The autocorrelation is non-summable. If the time series is wide sense stationary, the process becomes LRD.

We employ rescaled range (R/S) analysis and variance-time methods to calculate the Hurst parameter as a measure of long-term memory in time series [16]. Our results indicate strong LRD across all traces, with the Hurst parameter $H > 0.85$ at both small and large time scales. Notably, we do not observe obvious change points within the millisecond to second range, which differs from findings reported in previous studies [18], [19]. This is expected due to the increase in long-lived sessions such as video streaming and conferencing (Section II), as a percentage of overall backbone traffic, over the past decade. This finding also augments the evidence that the burstiness of the traffic is scale-invariant.

TABLE III: Characteristics of backbone link traffic

Trace	Period (JST)	Packets (M)	Avg Rate (Mbps)	Link Utilization (%)	Dispersion	Kurtosis	Skewness	Prominence
SF06	Mar 6, 23	108.73	871.77	87.18	3.38	5.11	0.46	72736
SF07	Mar 7, 23	105.41	900.06	90.01	3.27	5.85	0.05	89979
SF08	Mar 8, 23	104.59	788.01	78.80	3.50	7.87	0.59	97163
SF09	Mar 9, 23	115.02	839.18	83.92	3.56	7.22	0.47	85781
SF10	Mar 10, 23	116.78	895.41	89.54	3.55	7.29	0.32	111914
SF11	Mar 11, 23	53.60	324.19	32.42	2.99	12.63	1.11	64091
SF12	Mar 12, 23	52.88	309.43	30.94	3.25	13.82	1.03	63752

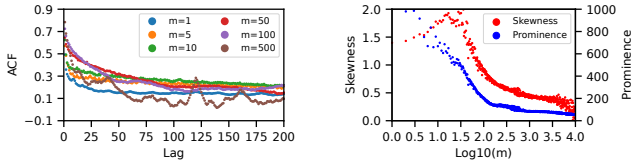

Fig. 3: Autocorrection coefficients (left) and skewness and prominence (right) computed over different time aggregation levels m (in ms) for trace SF09

Table III summarizes traffic characteristics of the seven traces we studied, calculated using the same methods and 1 s interval, as a comparison to Section II. Unsurprisingly, a much larger number of flows implies lower dispersion and kurtosis and higher prominence, compared to the single flow results in Tables I and II. We also analyze skewness and prominence, which are calculated based on packets per millisecond across different time aggregation levels in Fig. 3 (right). We find that both skewness and prominence values vary with time scale. This finding underscores the importance of selecting an appropriate time scale in routing path selection algorithms.

IV. Path Selection Approaches

Motivated by our observations above, we compare two simple approaches for balancing the load across multiple, equal-cost (ECMP) routing paths.

First, we experiment with the most straightforward policy: selecting the least loaded outgoing path for a new incoming long-lived application, such as a streaming or conferencing session. The simplest definition of *least loaded* uses a single load metric. We use path load, computed every time interval Δ. Load can be replaced by another metric, such as end-to-end path latency or packet loss. The algorithm can also randomize its choice among paths with "similar" loads. In our experiments, we use the following values for the time interval over which to compute load: $\Delta = \{0.05, 0.5, 1, 2, 5\}$ seconds.

Second, we again experiment with a least loaded policy, but we now use a slightly more complex load metric, based on a combination of the mean and variation of path load. Combining measures of mean and variation is a popular strategy that is used in the TCP protocol for retransmission timeout (RTO) computation [22]. Similar strategies are also used for estimating burst sizes for CPU scheduling.

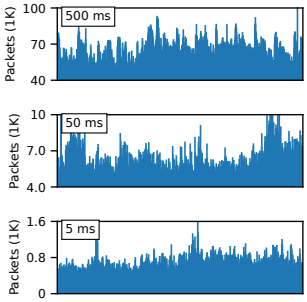

Fig. 2: Packets per time unit transferred at different time scales for trace SF09

TCP computes its RTO as the sum of a smoothed round-trip time (RTT) value and a factor (typically four) times a smoothed value of the deviation of the latest RTT sample from the smoothed RTT value [22]. Smoothed values are computed using exponentially weighted moving averages. We use a similar idea in Algorithm 1 to periodically update an estimate of the load value, *Load*, that we use in our least loaded policy. In our experiments, we set $\alpha = \frac{1}{8}$, $\beta = \frac{1}{4}$, $K = 4$, and vary Δ as discussed above. Note that the load value must be normalized if capacities are different.

Algorithm 1 Compute *Load*, given *CurrBytes*, the number of bytes transmitted during the past interval Δ.

Initialization: SBW $\leftarrow$ CurrBytes/Δ;
DevBW $\leftarrow$ SBW/2; PrevBytes $\leftarrow$ 0

1: DiffBytes $\leftarrow$ CurrBytes $-$ PrevBytes
2: CurrBW $\leftarrow$ DiffBytes/Δ
3: NewSBW $\leftarrow (1-\alpha) \times$ SBW $+ \alpha \times$ CurrBW
4: NewDevBW $= (1-\beta) \times$ DevBW $+ \beta \times |$SBW $-$ CurrBW$|$
5: SBW $\leftarrow$ NewSBW
6: DevBW $\leftarrow$ NewDevBW
7: PrevBytes $\leftarrow$ CurrBytes
8: Load $\leftarrow$ SBW $+ K \times$ DevBW

The inputs to the two algorithms, *e.g.*, counts of bytes sent per time period, are not difficult to obtain in practice. For instance, in Juniper Session Smart Routing [23], significant changes in path qualities can be conveyed to routers that make session routing decisions.

V. Experimental Setup

We conduct experiments using Mininet on a CloudLab [24] server of type rs630, which has an Intel Xeon E5-2660 processor with x86_64 architecture consisting of 40 CPUs with maximum speed of 2.6 GHz. The server runs 5.15.0-67-generic kernel with Ubuntu 22.04 and Mininet v2.3.1b1.

A. Topology

We use the simplest possible multipath topology: a topology with two parallel paths. Switch S1 acts as an ingress switch that can reach S4 through either intermediate switch S2 or intermediate switch S3. For unidirectional data traffic from

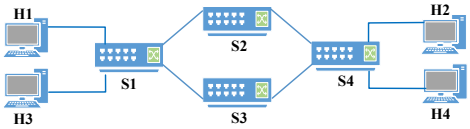

Fig. 4: Simple two-path topology

switch S1 to S4, paths S1-S2-S4 and S1-S3-S4 represent two equal-cost paths. Switch S1 executes the multipath routing algorithm. The bandwidth of links between hosts and switches is set to 100 Mbps with delay of 5 ms. The delay of inter-switch links is 20 ms and we vary the bandwidth below.

B. Applications

We consider the two types of applications we studied in Section II. An **adaptive video streaming** service involves a server (H3) that transmits a 200-second video using four-second video "chunks" to a client (H4) over TCP. The quality of each chunk is determined based on the buffer-based rate adaptation algorithm. We extend the DASH streaming client [25] to compute the QoE as given in [26] (Section V-D).[2]

A **video conferencing** service leverages the WebRTC [27] framework between hosts H3 and H4. Both users send a pre-recorded video of a talking head containing audio and video for a duration of 210 seconds. We extend the implementation in [28] to save the streamed video on H3 and the received video on H4 in order to compute QoE (Section V-D). Since Mininet creates different network namespaces and isolates the networks, STUN/TURN capabilities are restricted. Therefore, an independent TURN server is also configured on H3 so that media packets can be directly exchanged between H3 and H4.[3]

C. Background Traffic

From Sections II and III, we must experiment with different background traffic patterns. Host H1 acts as a sender and H2 acts as a receiver for UDP-based background traffic. We use ITG [29] to generate on/off traffic. The "on" period (t_{on}) represents the time during which traffic is sent with specified inter-departure times. No traffic is sent during "off" periods (t_{off}).

We compute the number of packets per second *(pps)* to send during an "on" period as $pps = \lfloor \frac{tput \times (t_{on} + t_{off})}{(psize \times 8 \times t_{on})} \rfloor$. We use an average throughput, *tput*, of 2 Mbps and a packet size, *psize*, of 1024 bytes in our experiments. Table IV lists the values of t_{on} and t_{off} in seconds (and the corresponding prominence values) for the sample background traffic patterns we use in the next section.

D. Evaluation Metrics

We consider both the perspective of the customer, who ultimately cares about application QoE, and the perspective of the ISP. We repeat each experiment five times and show the 95% confidence intervals for our results.

a) QoE Metrics: **Streaming QoE:** We use the QoE metric described in [26] which considers the video quality of a chunk, variation in chunk quality w.r.t. the previous chunk, rebuffering time, and startup delay. QoE is computed for each video chunk and normalized against the best possible QoE value. We average the normalized QoE for all chunks to obtain the QoE for a given session.

Conferencing QoE: We obtain sender and receiver videos of the WebRTC session in *webm* format. Frame numbers are extracted from both sender and receiver videos, and used for synchronization. We feed the raw videos to the libvmaf module of FFmpeg [30] to compare the sender and receiver videos, frame by frame, and compute the VMAF [10], PSNR, and SSIM [31] metrics, as was done in [32].

b) ISP Metrics: We examine link utilization, packet loss, packet end-to-end latency and jitter, since they impact the service-level agreements that the ISP can support, and the cost to the ISP. We also note router CPU and memory utilization, to ensure that path selection algorithms can be implemented on today's switches and routers.

TABLE IV: Background traffic patterns

Pattern	Streaming			Conferencing		
	t_{on}	t_{off}	Prominence	t_{on}	t_{off}	Prominence
1	2	0.5	512	3	4	1361
2	2	2	1165	2	4.4	1677
3	2	6	2012	0.8	4.8	2413

VI. Experimental Results

Our goals are to quantify the impact of background traffic on the QoE of video streaming and conferencing applications, and to gain a preliminary insight into the benefits of leveraging this information for multipath routing.

A. Impact of Background Traffic on Application QoE

In our experiments, the video streaming application bandwidth varies from 0.5 to 4.6 Mbps whereas the maximum bandwidth for the conferencing service is 9.5 Mbps. We systematically increase the link bandwidth between the switches S1-S2, S1-S3, S2-S4 and S3-S4, and observe application QoE.

We route two background traffic flows, *f1* and *f2*, with average bandwidths of 2 Mbps on links S1-S2 and S1-S3, respectively. Flow *f1* is an on/off flow as given in Table IV, whereas flow *f2* has constant packet inter-departure times.

The average bandwidth required to stream video at the highest quality while accommodating the background flow without any packet loss is 4.6 + 2 = 6.6 Mbps. Therefore, link bandwidth on the x-axis in Fig. 5 includes cases when the link is undersubscribed ($<$ 6.6 Mbps) or oversubscribed ($>$ 6.6 Mbps). Application QoE decreases as burstiness increases in Fig. 5. This behavior is observed even when the capacity exceeds the sum of average throughput values of the video streaming flow and background flow. For example, when

[2]Our changes to the python-based DASH player are available at https://github.com/UmakantKulkarni/dash-client.

[3]Our changes to the WebRTC framework are available at https://github.com/UmakantKulkarni/WebRTC-App.

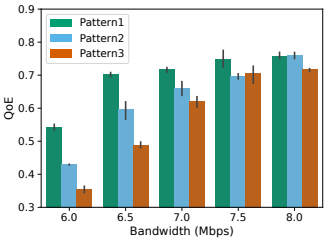

Fig. 5: Streaming QoE (ECMP)

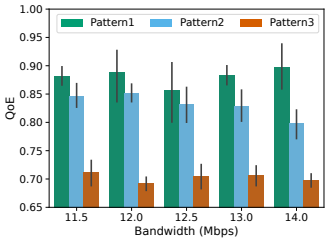

Fig. 6: Conferencing QoE (ECMP)

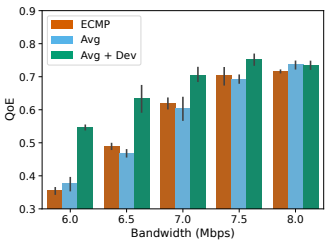

Fig. 7: Streaming QoE

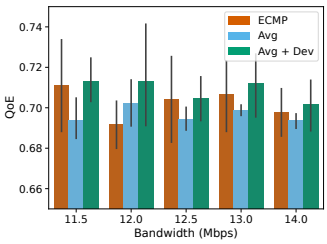

Fig. 8: Conferencing QoE

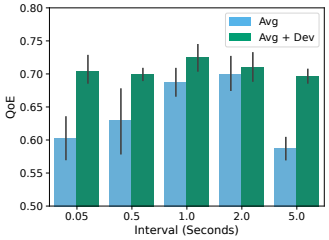

Fig. 9: QoE vs. Δ (streaming)

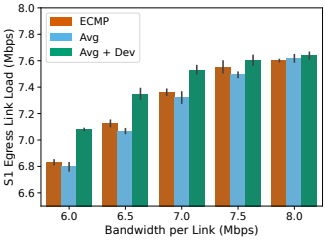

Fig. 10: Network load (streaming)

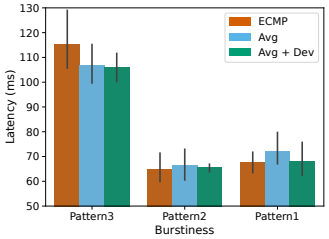

Fig. 11: Mean latency (conferencing)

TABLE V: Background traffic vs. QoE (conferencing)

Pattern	D	K	S	QoE
1	1047	1606	0.31	0.89
2	1351	1430	0.82	0.85
3	1693	1153	1.59	0.69

D = Dispersion, K = Kurtosis, S = Skewness

the bandwidth is 7, 7.5 or 8 Mbps, the QoE still degrades as burstiness increases.

The reason for the reduction in QoE lies behind how ECMP selects a path for each flow. ECMP hashes the 5-tuple of each incoming flow and uses round robin across equal-cost egress links. Therefore, background flows *f1* and *f2* use paths S1-S2 and S1-S3, respectively. The video streaming flow then uses S1-S2 which has the on/off background traffic. The TCP-based video streaming flow reduces its congestion window size during the "on" periods, reducing the QoE.

We also observe a difference in QoE in case of real-time video conferencing, as shown in Fig. 6. The real-time traffic uses RTP/UDP with the Google Congestion Control (GCC) algorithm [33], which adjusts the media sending rate when congestion is detected. The on/off background UDP flow causes the conferencing application to reduce its video resolution from the default 1920×1080.

B. Comparison of Path Selection Approaches

Fig. 7 and 8 depict the QoE of the streaming and conferencing applications with baseline ECMP, least-loaded based on a simple average, and least-loaded with Algorithm 1. We use background traffic pattern 3 and $\Delta = 0.05$ s in these experiments. For undersubscribed cases, the use of deviation in Algorithm 1 shows a clear benefit, especially for video streaming applications which are more bursty in nature (Table II and Fig. 1, right). Fig. 10 confirms that indeed more traffic can egress S1 in this case, reducing the cost to the ISP, which can now better utilize its links.

Fig. 9 examines the impact of the value of Δ on the two algorithms for the case when inter-switch link bandwidth is 7 Mbps. As expected, using a simple average is highly sensitive to the value of Δ since it does not consider traffic burstiness, which is observed in Internet traffic at multiple time scales as discussed in Section III. In contrast, Algorithm 1 yields a high streaming QoE for different values of Δ.

With Algorithm 1, the streaming and conferencing applications experience lower data-rate variation, thus sending their traffic at relatively high and stable bitrates. This reduces packet end-to-end latency and jitter (Fig. 11) and increases network utilization for undersubscribed cases (Fig. 10). This is desirable from the ISP perspective.

VII. Related Work

With the increasing deployment of Software-Defined Networks (SDNs), as well as Multi-Path TCP (MPTCP) and QUIC that enable applications to exploit multiple concurrent paths, there has been a resurgence of interest in traffic engineering and load balancing across multiple paths. For example, Hedera [5] is a centralized load balancing algorithm for data centers. CONGA [4] balances *flowlets* (bursts of packets followed by a relatively long period of inactivity) in data center networks based on collected congestion information. Weighted-ECMP (W-ECMP) [34] uses weights assigned based on active probing of links to determine the routing probability for each path in P4 switches. WCMP [6] is another weighted ECMP algorithm for tolerance to failures and changing topologies in data center networks.

Solutions such as CONGA [4] and W-ECMP [34] add methods to actively collect congestion information from a data center network. We only use passive measurements without active probing, and we do not restrict our work to data center networks. Additionally, most existing solutions [5], [34], [6] require SDN support. In contrast, our approach can be used with or without SDN, through integrating it with any switch or router, and any telemetry or measurement service. Our work also considers video streaming and conferencing QoE and traffic burstiness, on which prior work had not focused.

VIII. Conclusions and Future Work

In this paper, we analyze traffic burstiness of popular applications and backbone traffic at different time scales. Motivated by this analysis, we propose a routing path selection approach that uses both the average and variation of path load. We compare this approach to ECMP and simple average-based approaches in a variety of scenarios, and find that it shows promise in increasing application QoE and ISP resource utilization. We believe that this work serves as a first but important step in understanding the impact of background traffic characteristics and routing path selection on application QoE and resource usage.

We are currently experimenting with background traffic with different distributions of inter-packet times. Our preliminary results (Table V) show that certain characteristics of the background traffic time series can be proportional or inversely proportional to video conferencing QoE. As future work, we plan to explore machine learning models for predicting the path yielding the best QoE for a given application type. Additionally, we plan to conduct extensive experiments with real ISP topologies and settings.

References

[1] B. Fortz, J. Rexford, and M. Thorup, "Traffic engineering with traditional IP routing protocols," *IEEE Communications Magazine*, vol. 40, p. 118–124, 2002.

[2] D. Thaler and C. Hopps, "Multipath issues in unicast and multicast," RFC 2991, November 2000, http://www.ietf.org/rfc/rfc2991.txt.

[3] C. Hopps, "Analysis of an equal-cost multi-path algorithm," Request for Comments 2992, November 2000, http://www.ietf.org/rfc/rfc2992.txt.

[4] M. Alizadeh, T. Edsall, S. Dharmapurikar, R. Vaidyanathan, K. Chu, A. Fingerhut, V. T. Lam, F. Matus, R. Pan, N. Yadav, and G. Varghese, "CONGA: Distributed congestion-aware load balancing for datacenters," in *Proceedings of the 2014 ACM Conference on SIGCOMM*, ser. SIGCOMM '14, 2014, p. 503–514. [Online]. Available: https://doi.org/10.1145/2619239.2x626316

[5] M. Al-Fares, S. Radhakrishnan, B. Raghavan, N. Huang, and A. Vahdat, "Hedera: Dynamic flow scheduling for data center networks," in *Proc. NSDI*, 2010.

[6] J. Zhou, M. Tewari, M. Zhu, A. Kabbani, L. Poutievski, A. Singh, and A. Vahdat, "WCMP: Weighted cost multipathing for improved fairness in data centers," in *Proc. of the Ninth European Conference on Computer Systems*, ser. EuroSys '14, 2014. [Online]. Available: https://doi.org/10.1145/2592798.2592803

[7] F. Carpio, A. Engelmann, and A. Jukan, "DiffFlow: Differentiating short and long flows for load balancing in data center networks," in *2016 IEEE Global Communications Conference (GLOBECOM)*, 2016, pp. 1–6.

[8] K. MacMillan, T. Mangla, J. Saxon, and N. Feamster, "Measuring the performance and network utilization of popular video conferencing applications," in *Proceedings of the 21st ACM Internet Measurement Conference*, ser. IMC '21, 2021, p. 229–244. [Online]. Available: https://doi.org/10.1145/3487552.3487842

[9] G. Carofiglio, G. Grassi, E. Loparco, L. Muscariello, M. Papalini, and J. Samain, "Characterizing the relationship between application QoE and network QoS for real-time services," in *Proc. of the ACM SIGCOMM 2021 Workshop on Network-Application Integration*, 2021, p. 20–25. [Online]. Available: https://doi.org/10.1145/3472727.3472800

[10] Netflix, "VMAF - Video Multi-Method Assessment Fusion," https://github.com/Netflix/vmaf.

[11] D. L. Jagerman and B. Melamed, "Burstiness descriptors of traffic streams: Indices of dispersion and peakedness," in *Proceedings of the 1994 Conference on Information Sciences and Systems, Princeton, New Jersey*, vol. 1, 1994, p. 24–28.

[12] A. Marnerides, D. Pezaros, and D. Hutchison, "Internet traffic characterisation: Third-order statistics & higher-order spectra for precise traffic modelling," *Computer Networks*, vol. 134, pp. 183–201, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128618300677

[13] P. Varga, "Analyzing packet interarrival times distribution to detect network bottlenecks," in *EUNICE 2005: Networks and Applications Towards a Ubiquitously Connected World*, C. D. Kloos, A. Marín, and D. Larrabeiti, Eds. Boston, MA: Springer US, 2006, pp. 17–29.

[14] H. Haddadi, R. Landa, A. W. Moore, S. Bhatti, M. Rio, and X. Che, "Revisiting the issues on netflow sample and export performance," in *2008 Third International Conference on Communications and Networking in China*, 2008, pp. 442–446.

[15] R. Tao, J. Liu, Y. Song, R. Peng, D. Zhang, and J. Qiao, "Detection and optimization of traffic networks based on Voronoi diagram," *Discrete Dynamics in Nature and Society*, vol. 2021, p. 5550315, Dec 2021. [Online]. Available: https://doi.org/10.1155/2021/5550315

[16] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson, "On the self-similar nature of Ethernet traffic," *SIGCOMM Comput. Commun. Rev.*, vol. 23, no. 4, p. 183–193, oct 1993. [Online]. Available: https://doi.org/10.1145/167954.166255

[17] M. Crovella and A. Bestavros, "Self-similarity in World Wide Web traffic: evidence and possible causes," *IEEE/ACM Transactions on Networking*, vol. 5, no. 6, pp. 835–846, 1997.

[18] T. Karagiannis, M. Molle, and M. Faloutsos, "Long-range dependence ten years of Internet traffic modeling," *IEEE Internet Computing*, vol. 8, no. 5, pp. 57–64, 2004.

[19] H. Gupta, A. Mahanti, and V. J. Ribeiro, "Revisiting coexistence of poissonity and self-similarity in internet traffic," in *2009 IEEE International Symposium on Modeling, Analysis & Simulation of Computer and Telecommunication Systems*, 2009, pp. 1–10.

[20] T. Shapira and Y. Shavitt, "FlowPic: A generic representation for encrypted traffic classification and applications identification," *IEEE Trans. on Network and Service Management*, vol. 18, no. 2, pp. 1218–1232, 2021.

[21] K. M. Kenjiro Cho and A. Kato, "Traffic Data Repository at the WIDE Project," USENIX 2000 FREENIX Track, San Diego, CA, June 2000.

[22] M. Sargent, J. Chu, D. V. Paxson, and M. Allman, "Computing TCP's Retransmission Timer," RFC 6298, Jun. 2011. [Online]. Available: https://www.rfc-editor.org/info/rfc6298

[23] Juniper Networks, "Juniper session smart router," https://www.juniper.net/us/en/products/routers/session-smart-router.html.

[24] R. Ricci and Eide, E. et al., "Introducing CloudLab: Scientific infrastructure for advancing cloud architectures and applications," *The magazine of USENIX & SAGE*, vol. 39, no. 6, pp. 36–38, 2014.

[25] P. Juluri, V. Tamarapalli, and D. Medhi, "Sara: Segment aware rate adaptation algorithm for dynamic adaptive streaming over http," in *2015 IEEE International Conference on Communication Workshop (ICCW)*, 2015, pp. 1765–1770.

[26] X. Yin, A. Jindal, V. Sekar, and B. Sinopoli, "A control-theoretic approach for dynamic adaptive video streaming over http," *SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 4, p. 325–338, aug 2015. [Online]. Available: https://doi.org/10.1145/2829988.2787486

[27] "WebRTC," https://webrtc.org/.

[28] "WebRTC Samples," https://webrtc.github.io/samples/.

[29] D-ITG, "D-ITG, Distributed Internet Traffic Generator," https://traffic.comics.unina.it/software/ITG/index.php.

[30] "FFmpeg," https://ffmpeg.org/.

[31] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[32] B. García, F. Gortázar, M. Gallego, and A. Hines, "Assessment of QoE for video and audio in WebRTC applications using full-reference models," *Electronics*, vol. 9, no. 3, 2020. [Online]. Available: https://www.mdpi.com/2079-9292/9/3/462

[33] G. Carlucci, L. De Cicco, S. Holmer, and S. Mascolo, "Analysis and design of the google congestion control for web real-time communication (WebRTC)," in *Proc. of the 7th International Conference on Multimedia Systems*, 2016. [Online]. Available: https://doi.org/10.1145/2910017.2910605

[34] J.-L. Ye, C. Chen, and Y. Huang Chu, "A weighted ECMP load balancing scheme for data centers using P4 switches," in *IEEE 7th International Conf. on Cloud Networking (CloudNet)*, 2018, pp. 1–4.