Near-Real-Time Autonomous Multi-Path Flow Routing with Subflow Identification

H. Shakespear-Miles[1], N. Koneva[2], S. Barzegar[1], M. Ruiz[1], A. Sánchez-Macian[2], and L. Velasco[1*]

[1] *Optical Communications Group (GCO), Universitat Politecnica de Catalunya (UPC), Barcelona, Spain*
[2]*Dept. Ing. Telematica, Universidad Carlos III de Madrid, Spain - *luis.velasco@upc.edu*

Abstract: A distributed intelligence for autonomous near-real-time flow routing with subflow identification is proposed. Optimal subflow partitioning performed based on the subflow identification and the autonomously decided flow routing policy ensures continuous flow performance. © 2025 The Author(s)

1. Introduction

Packet-over-optical transport networks of the 6G era will need to support not only massive traffic volumes but also large traffic dynamicity. For this very reason, near-real-time operation is critical for ensuring network reliability and end-to-end (e2e) quality of service (QoS). In this context, machine learning (ML) techniques have demonstrated their effectiveness for improving resource utilization, introducing flexibility in optical networks, and enabling advancements in near-real-time autonomous network operation [1].

In our previous work in [2], we presented deep reinforcement learning (DRL) solutions for distributed autonomous packet flow routing in packet-over-optical networks, with the target to assure QoS (i.e., maximum e2e delay). The DRL solutions decided the fraction of *flow* traffic to be forwarded through a set of paths in near-real time. The set of paths was defined by the centralized software-defined networking (SDN) controller as part of the provisioning phase to provide enough aggregated capacity for the flow. Such architecture facilitates near-real time flow operation for QoS assurance, while liberating the SDN controller from such work. We assumed that every flow was splittable, i.e., it consisted of a number of subflows, so multipath flow routing could be performed by ensuring that packets belonging to the same subflow followed the same path in order to avoid packet reordering at the destination, which would increase delay for the application [3]. However, no mechanism for subflow identification was included. To solve subflow identification the Count-Min Sketch (CMS) probabilistic data structure can be used. CMS has shown its ability to identify the heaviest subflows in a packet flow (*heavy subflows* -HsF) and provide both an identifier and an estimate of their size. An implementation of CMS in P4 switches was experimentally assessed in [4]. In this paper, we extend our previous work in [1] and present an architecture for autonomous subflow traffic routing by integrating CMS for subflow identification running natively in the P4 switches. Besides, P4 switches can deal with complex routing tables, where subflows can be easily labelled and forwarded to the desired path, as well as implement in-band network telemetry (INT) to collect detailed end-to-end delay metrics [5]. We develop a new component that optimally selects the subflows that will be routed through each path based on the flow routing policy decided by the DRL engine (subflow partitioning). Additionally, we show how subflow identification can be used to give feedback to the SDN controller for the selection of the paths.

2. Network Scenario and Architecture

The proposed architecture and scenario are sketched in Fig. 1, where a traffic flow consisting of a large number of subflows enters the network at site A and it is forwarded to site B through 3 available paths ($p1$, $p2$, $p3$). The paths have been computed and set-up by the SDN controller to offer enough aggregated capacity for the flow. The P4 switch in site A forwards packets of each subflow through a path following the applied routing policies, while inserting INT-related fields. INT fields are collected by the P4 switch in site B and metrics (e.g., e2e delay) are computed and aggregated at short time intervals (e.g., hundreds of ms) to obtain flow telemetry measurements. With a larger periodicity (e.g., 1 second), the flow telemetry processor running on top of the P4 switch aggregates such measurements and sends them to the agent at the source site, where autonomous routing decisions are made.

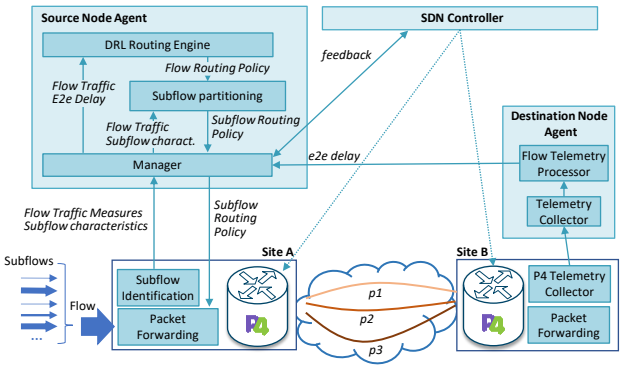

Fig. 1: Multipath flow routing with subflow identification

At site A, the *subflow identification* module runs inside the P4 switch and analyzes packets in real time to identify the heavy subflows and to estimate their bitrate. Note that a traffic flow can consist of a set of subflows that exist for a long period of time, and/or, a large number of subflows of very short bitrate and/or lifespan that cannot be individually identified. Still, we need to ensure that packets belonging to the same flow follow the same path. In

The research leading to these results has received funding from the European Union's Horizon Europe research and innovation programme SEASON (G.A. 101096120) and from the ICREA Institution.

consequence, the part of a traffic flow that cannot be identified is considered as a single (non-splittable) subflow.

At site A also, the *DRL routing engine*, running on top of the P4 switch, is in charge of determining the best flow routing policy, defined as the percentage of traffic to be sent to every path, to achieve the target QoS. Such decision is made periodically, e.g., every minute, based on flow traffic and e2e delay measurements, and processed by the *subflow partitioning* component that selects the sets of subflows to be forwarded through each path based on the flow traffic measurements and subflow identification received from the P4 switch. The objective of this procedure is to find the partitioning (denoted subflow routing policy) that better approaches the flow routing policy provided by the DRL routing engine. The subflow routing policy might lead to traffic *volumes* to each path that significantly differ from the continuous and ideal flow routing policy. That happens when the traffic flow includes some subflows with large traffic volumes, which might make it impossible to find partitions that accurately follow the routing policy received from the DRL engine, resulting in excessive delay for some of the subflows and in flow QoS violation. Hence, it is of paramount importance to analyze the flow composition (which might change with time), as well as the error between the volumes of the subflow partitions and the routing policy. Algorithms in the *manager* analyze the performance of the operation and give feedback to the SDN controller so as it can dynamically update the paths available for the flow operation.

3. Models and Algorithms

In this section, we detail the main components defined in the previous section. A CMS is a probabilistic data structure used to approximate frequencies of events (packets) on a stream of data. A CMS is dimensioned by parameters ε and δ, where ε is the maximum relative error and δ is the probability of exceeding the maximum error. The data structure is a matrix with d rows and W columns. Associated to the matrix, d hash functions are evaluated for every incoming packet. Then, given a number k of packets, the upper bound on the estimation error can be expressed as $\widehat{C_k} - C_k \leq \varepsilon \cdot k$, where $\widehat{C_k}$ is the estimation from the CMS and C_k is the true count. Note that such error is directly related to the estimation of the traffic of the HsF. Please, refer to [4] for further details.

The DRL engine consists of an agent and an environment to determine the flow routing policy. The agent is responsible for learning the best action to be taken based on the current routing state and reward, and the environment computes the state and reward and shares it with the agent periodically. The DRL engine implements a Twin Delayed Deep Deterministic Policy Gradient (TD3) off-policy technique. We showed in [1] that the QoS required for the flow is achieved by implementing the DRL flow routing policy.

The subflow partitioning algorithm is defined in Algorithm 1. The algorithm receives the aggregated flow traffic currently observed ($X(t)$), the flow routing policy (P) with the IDs of the paths available and the percentage of traffic that should be routed thought each one, and the identification and the estimated traffic for each HsF observed (H). Because the routing policy will be applied to the incoming flow traffic for the next time interval, the algorithm uses a simple polynomial predictor to predict the total traffic during the next time interval (line 1). The prediction is used to scale the estimated HsF traffic proportionally (line 2). The non-HsF traffic is then calculated (line 3) and stored (line 4). Then, the subflows are sorted by descending order so that those with larger traffic requirements are handled first (line 5). Each subflow is then checked to find the path that would result in the smallest capacity difference without exceeding that given by the routing policy, considering the subflows already evaluated, and the selection is saved (lines 6-9). The resulting subflow routing policy is eventually returned (line 10).

Algorithm 1: Subflow partitioning algorithm.

Input: $X(t)$, P, H **Output**: sFR

1: $X(t+1) \leftarrow$ predict($X(t)$)
2: **for each** $id \in H$ **do** $H[id] \leftarrow H[id] \times (X(t+1)/X(t))>$
3: $nH \leftarrow X(t+1)$ - sum(H)
4: $H[0] \leftarrow nH$
5: $H \leftarrow$ sort(H, DESC)
6: $sFR \leftarrow \{\}$
7: **for each** $id \in H$ **do**
8: $path\_id \leftarrow$ find_min_diff(sFR, P, H)
9: $sFR \leftarrow sFR \cup \{<id, H[id], path\_id>\}$
10: **return** sFR

4. Illustrative Results and Conclusions

A Python-based network simulator has been used to evaluate the proposed solution. The simulator includes a traffic generator with two components that create packet traces for the different subflows, where the total flow traffic follows a daily pattern. First, a time series defining the traffic volume for each of the subflows is generated randomly based on configuration parameters. Then, packet traces are generated for each subflow. Packet size is chosen according to a trimodal distribution with values [64, 596, 1500] and probabilities [0.333, 0.167, 0.5], whereas inter-arrival times are adjusted for each subflow in order to approach as much as possible the subflow time series values. A total of 16 packet traffic scenarios were generated, each based on different combinations of flow rates, packet rates, and the proportion of total traffic identified as HsF. These scenarios were created following the traffic statistics in [4], combining four total average traffic rates—10, 70, 160, and 280 Gb/s—with three HsF traffic proportions—50%, 60%, and 90%.

Let us first evaluate the accuracy of the CMS for subflow identification. For these tests, we configured a monitoring time window of 10 ms and considered scenarios with 20 HsFs, the rest of the traffic being a large collection of very small subflows. The average packet count error is plotted in Fig. 2 for several configurations in terms of d and W values. Specifically, we considered W=55 (Fig. 2a-c) and W=256 (Fig. 2d-f). Overall, W=256 provides accurate packet count, while $d \geq 5$ hash functions minimizes the count error.

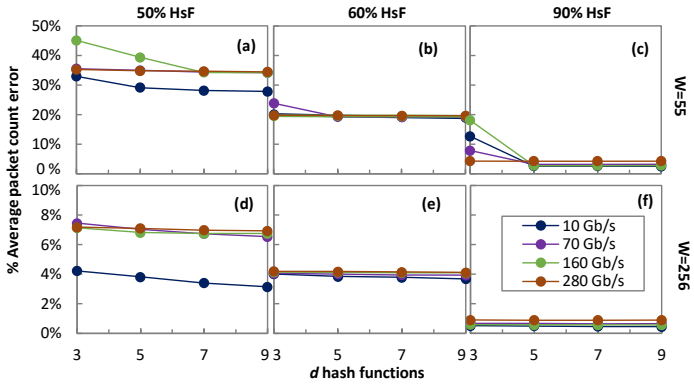

Fig. 2: Error during subflow identification

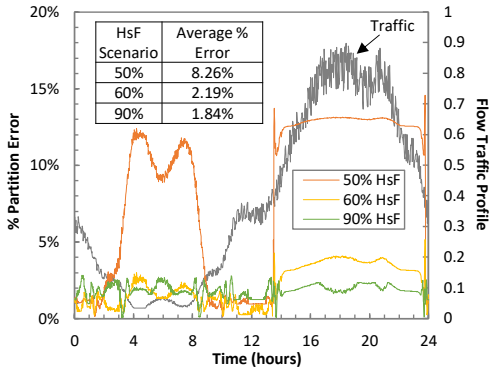

Fig. 3: Error during subflow partitioning

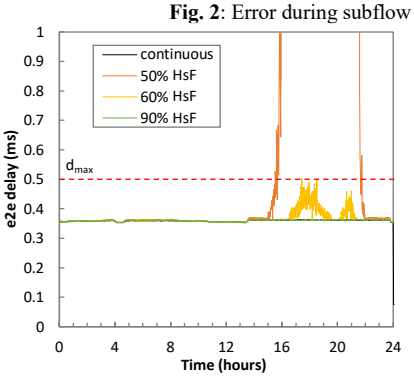

Fig. 4: Delay with subflow partitioning

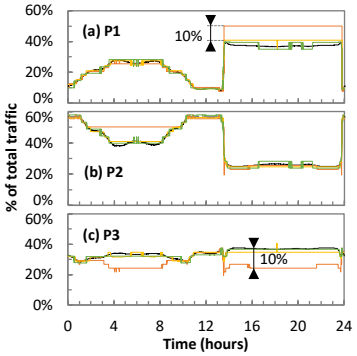

Fig. 5: Flow traffic through the three paths

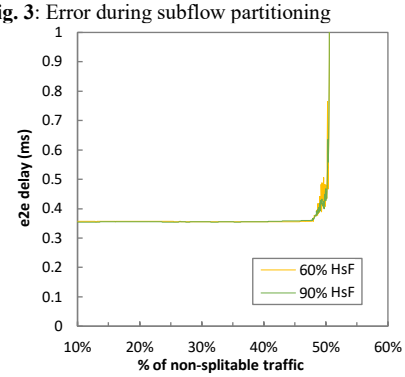

Fig. 6: Delay with increasing single HsF

We also observe in Fig. 2 that the percentage of HsF traffic w.r.t. the total flow traffic has a larger impact on the error than the maximum traffic on subflow identification, so we concentrate on a single scenario with maximum average traffic of 160 Gb/s following the daily pattern in Fig. 3. Three paths were configured with the following available capacity in Gb/s and in % w.r.t. the peak of traffic: p1: 90 Gb/s (56%), p2: 60 Gb/s (38%), and p3: 67 Gb/s (42%). Note that the aggregated capacity of the paths is 217 Gb/s, enough to support the peak of traffic and with no significant impact on the e2e delay if traffic is routed through the paths accordingly. In addition, a pretrained TD3 model for 3 paths was used for the DLT routing engine.

Let us first evaluate the performance of the subflow partitioning algorithm to provide accurate subflow routing policies. The algorithm implemented a polynomial traffic predictor of degree 2 with a window size of 10 to estimate the total aggregated traffic for the next time period. Fig. 3 shows the partitioning error as a percentage of the total traffic for each proportion of HsF. We observe that the error is under 2% on average for the 90% HsF scenario and increases to around 2.2% for the 60% HsF scenario. However, a large error of over 8% with peaks of around 13% can be observed for the 50% HsF scenario, which concentrate during low and high traffic periods.

The impact of subflow partitioning on the e2e delay along a day is plotted in Fig. 4 for the considered HsF scenarios and together with delay obtained by executing the flow routing policy from the DRL engine (labeled *continuous*). We observe that the combination of partitioning error and high traffic period translate into large increments of delay, which become largely excessive when the non-splittable traffic volume gets closer to the capacity available in the paths. The effect of the error is clearly observed in Fig. 5. In the 50% HsF scenario, 50% of the traffic could not be identified and was labeled together as non-splittable. Consequently, routing policies from the DRL engine (e.g., [40, 30, 30]% on paths p1..p3) are translated to policies with at least 50% on one of the paths (e.g., [50, 30, 20]%). Note that to minimize the delay, the DRL engine selects policies that are far enough from saturating the capacity of the paths. To clearly identify the maximum non-splittable traffic that can be handled with the three paths available, let us consider a constant traffic profile along the day and vary the percentage of one of the HsFs for the 60% and 90% HsF scenarios. Fig. 6 shows the obtained e2e delays, which are coincident for both scenarios and show that when the traffic of the subflow approaches the maximum capacity of the available paths, the delay increases sharply. To avoid that, paths with capacity matching HsF composition are needed.

In conclusion, we have shown the importance of accurate subflow identification for the autonomous multi-path flow routing to assure flow QoS near-real-time. HsF identification and partitioning ensure that subflow packets follow the same path, thus avoiding packet reordering at destination. In addition, estimates from HsF identification can be used by the SDN controller to provide paths with capacity matching the specific flow traffic characteristics.

References

[1] D. Rafique and L. Velasco, "Machine Learning for Optical Network Automation: Overview, Architecture and Applications," JOCN, 2018.
[2] S. Barzegar *et al*., "Autonomous Flow Routing for Near Real Time Quality of Service Assurance," TNSM 2024.
[3] J. Lin *et al.*, "Packet Reordering in the Era of 6G: Techniques, Challenges, and Applications," Electronics, 2023.
[4] J.A. Hernandez *et al*. "Count Min sketches for Telemetry analysis of performance in P4 implementations," ONDM 2024.
[5] P. Gonzalez *et al*., "Distributed Multi-Agent System fed with Telemetry Data for Near-Real-Time Service Operation," OFC, 2024.