Title: Real-time Control and Management Plane for Edge-Cloud Deterministic and Dynamic Networks

Authors: Mijail Szczerban,Nihel Benzaoui,Jose Estaran Tolosa,Haik MARDOYAN,Achour Ouslimani,Abed-elhak Kasbari,Sebastien Bigo,Yvan Pointurier

Accepted: 22 July 20

Posted 22 July 20

DOI: https://doi.org/10.1364/JOCN.397020

Real-time Control and Management Plane for Edge-Cloud Deterministic and Dynamic Networks

Mijail Szczerban[1,2,*], Nihel Benzaoui[1], José Estarán[1], Haïk Mardoyan[1], Achour Ouslimani[2], Abed-Elhak Kasbari[2], Sébastien Bigo[1], and Yvan Pointurier[1”]

[1] *Nokia Bell Labs, 7 route de Villejust, Nozay 91120 France*
[2] *ENSEA, 6 Avenue du Ponceau, 95000 Cergy, France*
**mijail.szczerban_gonzalez@nokia.com*

Received XX Month XXXX; revised XX Month, XXXX; accepted XX Month XXXX; posted XX Month XXXX (Doc. ID XXXXX); published XX Month XXXX

Today's optical networks dynamicity is far from its potential. Optical components, such as fast-tunable lasers or semiconductor optical amplifiers, can react in nanosecond time scale, while optical networks reconfiguration time is many orders of magnitude larger, normally above hundreds of milliseconds time scale. In this work, we address this gap with real-time control plane strategies that enhance the responsiveness of optical networks, specifically in the context of time-critical applications where service determinism is of paramount importance. This context represents an additional challenge since the infrastructure necessary to provide time-wise guarantees increases the complexity of the system under control. We describe in detail the real-time control plane for deterministic and dynamic networks and assess its value through experimental evaluation for the first time of a complete real-time control plane within a multi-network segment testbed. We prove sub-millisecond overall reconfiguration time for multi-network segment environments spanning distances in the order of tens of kilometers.

http://dx.doi.org/10.1364/JOCN.99.099999

1. Introduction

Applications such as 5G front-haul, industry 4.0, high-frequency trading, and telesurgery demand strict time-wise performance guarantees [1] [2] [3]. Time-wise performance relevant metrics are latency (source-destination frame delay), jitter (latency variability), and network reconfiguration delay. Most demanding time-critical applications typically expect tens to hundreds of microseconds latency, sub-microsecond jitter, and millisecond service turn-up time [4]. Note that these requirements span two different dimensions: first, performance determinism with a pre-defined latency featuring low variability (jitter); and second, dynamicity, expressed by fast reconfiguration. Furthermore, cloud services can be established between distant endpoints, e.g., servers located in data center premises and clients in access networks (data centers and access networks being different network segments), thus, end-to-end guarantees over multi-segment networks and fast reconfiguration are needed.

Currently, no solution provides both end-to-end determinism and ultra-dynamic reconfiguration simultaneously. Determinism and dynamicity are two forces pulling in opposite directions. To achieve determinism, network resources need to be reserved from source to destination, typically a complex and time-consuming process because there can be potentially many elements to be configured using different control plane protocols, thus, playing against network dynamics.

Therefore, we would find high value in networks that could provide time-wise determinism while being highly dynamic (below millisecond scale reconfiguration) to allow for incoming services to be deployed seamlessly. The challenge addressed in this paper is to enable performance guarantees as if each flow was sent over a dedicated fiber while reconfiguring fast enough to support most dynamic applications. In this work, we achieve ultra-high dynamics by revisiting control plane components and optimizing their response time, down to sub µs contributions.

A. Available time-sensitive solutions

Optical communication technologies can address time-sensitive services in different ways. First, some solutions can provide deterministic performance by creating independent physical or logical communication channels through the reservation of transmission resources (WDM and TDM). This is the case of technologies such as OTN, FlexE, and standard PON. Nonetheless, these technologies feature quasi-static configuration, failing to provide fast dynamics, typically above seconds time scale [1]. Second, other communication solutions provide quality of service (QoS) strategies to improve time performance. IEEE 802.1Qbu and 802.1Qbv Time-Sensitive Networking (TSN), standards [5], improve time performance with respect to plain Ethernet using frame preemption and class-of-service differentiation, a priority-driven policy that makes each class experience different performance. However, in [4], we showed that TSN class-based approach fails to deliver deterministic and low latency when many flows with equal priority coexist. We require a different approach, a solution allowing flow granular resource reservation to guarantee determinism and a control plane that enables end-to-end real-time network reconfiguration to establish flows on-demand.

B. Optical networks control plane

Optical network control plane defines the routing policies and manages (establishes or releases) optical connections. Control plane automation

[1“] *Yvan Pointurier is now with Huawei Technologies, France.*

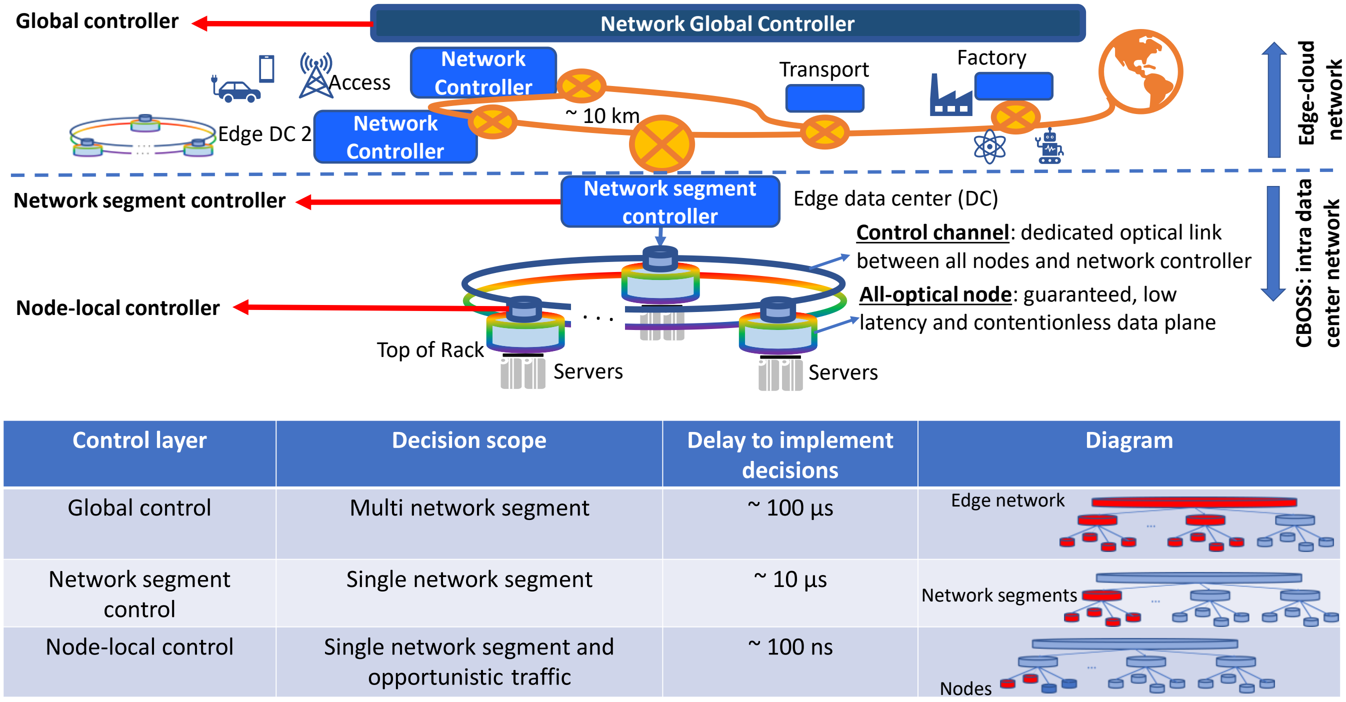
is a very important topic for network operators which can capture operational benefits from a common management infrastructure [6]. Standardization bodies, such as the Internet Engineering Task Force (IETF) and the International Telecommunications Union-Telecom (ITU-T), have defined control plane protocols such as GMPLS and ASON, providing a framework for optical network elements interaction and configuration [7] [8] [9]. Extensive work has been carried out to achieve the virtualization and programmability of networks, Software-Defined Networking (SDN) proposals centralize network control, emphasizing the role of software on network management. This allows to make an abstraction of network physical resources, increasing modularity, and enabling technology-agnostic network control. SDN provides network programmability via open interfaces, which is of great interest to ease network management. Nonetheless, SDN does not provide directives on time-wise optimization of the control plane reconfiguration [10] [11]. Network dynamicity can be affected by strictly unnecessary protocols, functional layers, and decision centralization [12]. In practice, the utilization of slow interfaces between control plane decision-making units and data plane devices, as well as the use of general-purpose CPUs as control element, subject to interruptions and time sharing between applications, makes current SDN-based network reconfiguration slow, typically providing response time over hundreds of milliseconds time scale. Work has been carried out to make SDN compatible with time-sensitive applications, for instance, time-aware enhanced SDN (TeSDN) introduces a resource scheduling scheme for time-sensitive services over elastic optical networks [13]. TeSDN can effectively improve application performance by adjusting the network and application resource allocation according to the QoS requirements. Nonetheless, TeSDN is based on OpenFlow, thus, control plane delays are not tightly controlled since it uses non-deterministic communication protocols and varying processing time for monitoring and resource allocation.

In [14], they experimentally evaluate all-optical dynamic networks supporting Virtual Network Functions (VNF) and providing some degree of service guarantee, featuring a reconfiguration time in the order of hundreds of milliseconds. Additionally, the complete data center function virtualization (including computing resources) in all-optical data planes environment has been implemented using non-real time SDN [15]. In [16], intelligent optical tunnels are proposed to deal with edge-cloud requirements, using both pre-allocation and dynamic allocation of resource, reporting reconfiguration over the millisecond time scale. In [17], a flat architecture is proposed to reduce latency and increase scalability providing SDN-based dynamic reconfiguration time over the hundreds of millisecond time scale for data center networks [18]. Research work has achieved end-to-end control plane integration of existing time-sensitive solutions, specifically TSN and SDN-based networks, the so-called Time-Sensitive Software Defined Network [19]. Nonetheless, its reconfiguration time is in hundreds of milliseconds order, even when using simplified scheduling, due to delays of intrinsic protocols used, and only allows for time-sensitive services in TSN network segments.

The utilization of real-time reconfigurable hardware to enhance cloud data center computing capacity has been studied and implemented in [20], showing to be a valuable and mature technology increasing reconfigurability and data center resource utilization. Overall, although most of these research works acknowledge the importance of high dynamics for enhanced performance, they provide dynamics (reconfiguration) over the milliseconds time scale at best and, in most cases, even over hundreds of milliseconds.

C. Deterministic and Dynamic Networks

Deterministic and Dynamic Networks (*DDN*) [4] raises a future-proof optical solution, guaranteeing time-wise performance while being responsive enough to adapt to new network conditions and establish new flows seamlessly. *DDN* is based on the following characteristics: (*i*) contentionless data plane leveraging flow-granular resource allocation and providing predictable performance; and, (*ii*) fast control plane enabling ultra-fast and reliable monitoring, decision making, and resource allocation. *DDN* can bring determinism thanks to per-flow network slicing and a jitter compensation mechanism [4]. Nonetheless, determinism requires a complex networking infrastructure, raising concerns about its dynamicity. In [21], we explored the dynamicity of DDN in intra-data center network based on a centralized real-time controller. This controller enables a network segment-wide reconfiguration in tens of µs order, for intra-data center network with



Global controller
Network Global Controller
Network Controller
Access
Transport
Factory
~ 10 km
Edge DC 2
Network Controller
Edge-cloud network
Network segment controller
Network segment controller
Edge data center (DC)
Control channel: dedicated optical link between all nodes and network controller
All-optical node: guaranteed, low latency and contentionless data plane
Node-local controller
Top of Rack
Servers
Servers
CBOSS: intra data center network

Control layer	Decision scope	Delay to implement decisions	Diagram
Global control	Multi network segment	~ 100 µs	Edge network
Network segment control	Single network segment	~ 10 µs	Network segments
Node-local control	Single network segment and opportunistic traffic	~ 100 ns	Nodes

Fig. 1: Edge-cloud network scheme showing real-time control layers and CBOSS as intra data center network segment

3.3 km transmission distance. In [22], we introduced the node-local control layer, with tens of nanoseconds time scale local decisions. This enabled features such as opportunistic traffic insertion.

To achieve high network dynamicity, we propose and implement a real-time control plane for optical networks that increases the dynamicity of the network through the optimization of control and management infrastructure from the hardware level. In DDN environment, these goals must be accomplished while guaranteeing data plane deterministic performance. Fig. 1 shows the schematic diagram of our real-time control architecture in an edge-cloud network environment. Edge-cloud architecture aims at reducing latency (propagation delay) for time-sensitive applications by bringing computing capacity into distributed small size data centers closer to the end user [4].

In this work, we further explain the concept of real-time control, how it differentiates against current solutions, propose, implement and experimentally evaluate for the first time the real-time global control layer to assess its dynamicity for multi-segment network environments. In Section 2, we define real-time control plane strategies to increase the reactiveness of the network. In section 3, we explain the experimental proof of concept and the components enabling both deterministic performance and ultra-dynamic features. Section 4 presents the experimental results, including the emulation of distributed processing showcasing the value of high dynamics for time-sensitive applications. Section 5 concludes the paper and summarizes the main results.

2. Real-time control plane

We consider control and management function as a continuum [23], and we refer to these functions as real-time control plane when the strategies described in this section are applied. Current optical elements such as fast-tunable lasers and semiconductor optical amplifiers (SOAs) can provide nanosecond time scale reconfiguration [24] [25]. In contrast, state of the art control plane response time is, at best, in the millisecond order, as described in the previous section. There is an important gap between optical data plane reconfiguration potential and current control plane dynamicity. Some of the causes of this gap are monitoring/control instruction propagation delay, the use of slow interfaces between control entities, slow processing due to general-purpose CPU (non-deterministic), and resource scheduling decision complexity. The goal of the real-time control plane is to reduce the existing gap between optical elements and network reactiveness.

To improve network's dynamicity, it is necessary to revisit all control plane components and optimize their response time, down to sub µs contributions to reach optical networks dynamic potential. This implies that the solution cannot be only software-based, we require to re-design the control plane from its physical infrastructure. Thus, our real-time control plane breaks with typical SDN paradigm that promotes the abstraction of the underlying physical infrastructure. The real-time control plane needs strict integration with the underlying physical infrastructure to achieve the maximum dynamicity.

There are three fundamental network control functions that require enhancement to achieve real-time control plane performance: *i)* monitoring system to assess the state of the network, *ii)* real-time decision-making to react and decide promptly on the allocation of resources, and *iii)* network instruction execution mechanism that allows implementing decisions rapidly.

Regarding control plane physical layer elements, our real-time strategy increases network reactiveness through dedicated and high-speed (>10Gb/s) communication between all control entities, with transmission time slots reserved per-control function and per-networking element, ensuring deterministic delays. For decision-making units (controllers), we implement real-time and tailored processing, through application-specific integrated circuits (ASIC) or field-programmable gate arrays (FPGA). These processing units provide hardware adapted to required functions and support high-speed and parallel processing. The FPGA can be of interest since it allows for the reconfiguration of hardware implementing logic functions. The FPGA allows reconfiguring the control plane and the networking element itself from the physical layer, enabling a hardware-defined network. Regular SDN architecture imposes the separation of the controller and the controlled entities, requiring an open interface for communication [10]. Our real-time control plane strategy, in the other hand, promotes the integration of the controller and controlled entities at all possible levels to avoid unnecessary delays.

Propagation delay is a physical constraint and large networks are inexorably prone to it, this is why edge-cloud network approach has been selected for the DDN proposal of this work, by having small-sized edge data center covering a radius of few tens of kilometers and implementing a distributed control approach that brings network controllers as close as possible to the network elements. From a control plane architecture perspective, our real-time control plane strategy consists on a decision-making scope split in different layers – regarding the allocation of communication resources (electronic and photonic) in specific network segments – to reduce the decision complexity and propagation delay. Decisions are divided into smaller –simpler– pieces, enabling faster processing, and decision-making entities are brought closer to the element under control, thus, reducing propagation delay for control, management and monitoring functions. We define three real-time decision-making layers, see Fig. 1:

Global control layer: oversees services spanning multiple network segments, e.g. inter-data center communication. It is embodied by a real-time global controller embedded in the network orchestrator that serves as mediator between all network segment controllers to establish end-to-end flows while guaranteeing service performance. The real-time orchestrator is the entity with the higher perspective on communication and computing resource availability across the entire edge cloud network. Assuming few tens of kilometers edge network, the global control layer provides responsiveness in the order of hundreds of µs, mainly dominated by propagation delay.

Network segment local control layer: this layer manages a single network segment through a real-time centralized network controller that retrieves monitoring information from all nodes on the state of transmission resources utilization, per-flow latency, flow queue filling, etc. The network segment controller takes decisions considering real-time monitoring information and decides on the allocation of resources accordingly. It also handles orchestrator requests to guarantee inter-segment services. Decisions whose scope is limited to a single network segment, e.g. intra-data center (intra-DC) virtual machine migration, can be autonomously executed by this layer. Assuming few kilometers network segments, the responsiveness of this layer is in the order of tens of µs dominated by propagation. By design, the network segment controller is located as close as possible to the networking elements; it cannot be remote since that would increase propagation delay.

Node-local control layer: the main contribution to the delay incurred by the network control plane is due to propagation delay, light speed in fiber is around 5 µs/ km, which even in an edge-cloud context becomes considerable. To provide a control plane that can react at the speed of optical elements as described earlier (sub µs), it is mandatory to create a decision-making layer collocated to the element it controls. To embody the real-time node-local control layer, a high-speed processing unit (ASIC or FPGA) should be integrated into the node, and high-speed interfaces must be enabled exclusively for control plane purposes. This might not be feasible with most of commercially available devices, nonetheless the adoption of FPGA into the network switching fabric is gaining momentum [20] [26]. Wavelength insertion and client data queue reading are examples of decisions that can be taken by the node with proper signaling mechanisms. Node-local control layer can manage resources that are local to the node, taking decisions on a per optical time slot (~µs) basis. The node local controller leverages resource utilization flags (wavelength and time slot) to avoid interferences with other transmissions. This control layer is the fastest since decisions are directly taken by the node real-time controller and no propagation is required. The decision time scale of this layer is in the order of few ns to few tens of ns depending on the controller clock rate and decision complexity. The mechanisms used by this control layer are further explained in the proof of concept section.

Overall, our solution differs from other proposals such as TeSDN [13], in that we provide data plane performance guarantees but in addition the control plane infrastructure is designed to be highly reactive and predictable (deterministic).

3. Proof of concept: real-time control plane for deterministic and dynamic edge-cloud networks

To validate the compliance of our real-time control plane proposal with the dynamicity requirements of *DDN*-class networks [4], we implement and experimentally study the strategies described in the previous section on an edge-cloud network prototype.

A. Intra-data center network segment

We leverage Cloud-Burst Optical Slot Switching (CBOSS) [27], which is an all-optical slot switching intra-data center (DC) network architecture, to evaluate the real-time control concept in intra-DC networks. CBOSS relies on a transparent data plane avoiding any packet contention in intermediate nodes, see Fig. 2. It leverages wavelength (λ) and high-granular time-division multiplexing (~µs optical time slots), which in combination with dedicated queues and interfaces, enable per-flow network slicing which essential to provide time determinism in the network. Concerning the control plane, CBOSS features a dedicated optical control channel, which is systematically dropped, processed and retransmitted at each node, providing a guaranteed path to transmit control and management information including routing instructions, monitoring data, network's synchronization, and transmission schedule. The control channel uses optical O-band, whereas data channels use C band, providing a physical split between both channels. Transmission schedule informs which flow (λ and queue) to be served at each time slot. The intra-data center testbed used on the following experiments consists of a 3-node real-time CBOSS ring prototype [27]: one Master linked to the network controller and two Slaves. Each node equipped with a fixed-wavelength optical 10G receiver, and for the data plane a fast-tunable (ns scale) WDM 10G transmitter, using arrays of C-band DWDM SFP+ and SOAs as gates. The total ring length is ~3.5km of SMF, with equally split inter-node links. Fig. 2, shows the node scheme, including optical paths of control and data channels, the FPGA node controller and the WDM transmitter. On top of Fig. 2 there is an example of the optical data plane WDM scheduled transmission for two wavelengths and at the output of the transmitter, during a 10 time-slot periodic reservation. Note that some slots can be empty (no power) and others can have more than one λ inserted.

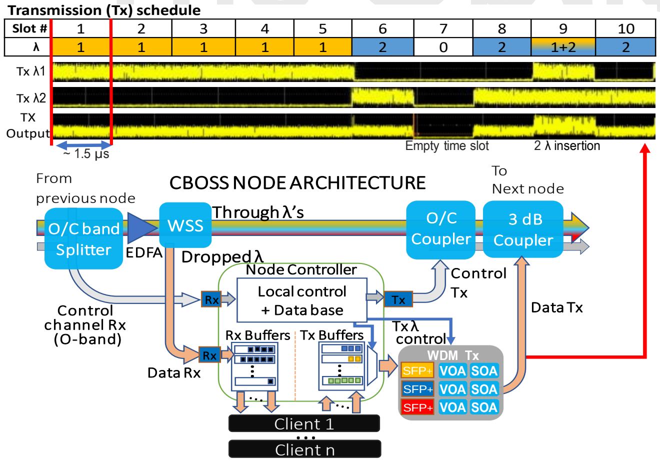


Fig. 2: CBOSS node architecture with WDM transmission schedule example on the top.

CBOSS architecture presents three fundamental characteristics that makes it a *DDN*-capable solution: first, the transparent data plane that prevents contention (and corresponding delay) at intermediate nodes; second, the fast-time slotted switching that permits to enforce the resource reservation with high granularity, enabling per flow network slicing; third, the dedicated and high-speed control channel in combination with the real-time control unit, allowing to monitor the state of the network and deliver control instructions promptly and predictably. CBOSS is presented as a plausible solution but not the only way of implementing *DDN*. Other network architectures can be designed to be adapted for *DDN* as long as they support a deterministic data plane and high-speed control plane. We acknowledge that the utilization of real-time processing units and dedicated high-speed control channel can lead to additional costs, but deterministic applications can be enabled through this infrastructure, adding high value to the network. The solution presented in this work is designed for small-sized edge-cloud networks, with limited scalability.

B. Inter-data center/transport network segment

We built a 2-node inter-data center (inter-DC) network segment, based on the same node controllers as CBOSS, thus, providing real-time control plane as well. To emulate an edge-cloud environment, the nodes were separated by few kilometers of fiber in both data and control plane. The communication between both network segments was performed through opaque 10GE gateways interfaces, thus, both network segments could operate asynchronously.

C. Real-time control plane implementation

Control and management functionalities such as topology discovery, frame route selection, fault detection, monitoring, resource commissioning and provisioning as well as network reconfiguration, were implemented for the proof of concept. The three control layers have been implemented:

1. *Global controller*

For the proof of concept, we implement for the first time the real-time global controller on an FPGA (Xilinx Kintex UltraScale evaluation board). The global controller establishes high-speed bidirectional communication (>10Gb/s) with both network segment controllers (intra-data center and transport). It also enables high-speed communication to enable the reception of service requests from external sources. The global controller FPGA logic can be programmed to react depending on the inputs coming from external service requests and also requests coming from lower control layers (segment controllers). For inter-segment connection establishment, the request can come either by an external source or from lower control layers (segment controllers). The requests are processed by the global controller, that decides which network segments are needed to establish the communication. The global controller transmits requests to the network-segment controllers which can allocate network resources accordingly.

2. *Network segment controller*

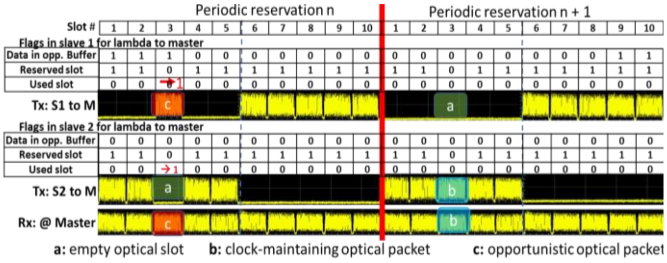
The FPGA-based network segment controller monitors the network in real-time, receiving information from nodes on the state of the optical signal, queue filling, flow latency, client count, etc. It takes decisions on the transmission schedule to be implemented, indicating the wavelength and client data queue to read at each time slot by each node to avoid interferences between different node transmission and provide service guarantees (latency, jitter, and transmission capacity). The network controller establishes high-speed communication with the global controller and with a special node in the network segment – Master node – that serves as an interface with all network nodes. Control instructions are retrieved by the Master node and transmitted to the corresponding nodes through the dedicated high-speed control channel that connects all network nodes. Likewise, monitoring information is sent from all nodes to the Master node, which in return transmits this information to the network segment controller. Different nodes can retrieve the same control instructions in different time due to distinct propagation delays. In our experimental solution we solve this mismatch by synchronizing the transmissions schedule updates through the control channel that travels in parallel with the data channels. The implementation of new transmission schedule in the data plane "follows" the reference given by the control channel and the new schedule is implemented in the next time slot after the schedule is retrieved at each node.

In a general case, there can be more than one node communicating to the network controller for enhanced reliability; nonetheless, for this proof of concept, we use a single Master node per network segment.

3. Node-local controller

Each network node is composed of a node controller (FPGA) in charge of managing physical layer resources, including optical switching, medium-access control, client interfaces and data queues, as depicted in Fig.2. The node controller prior to the real-time control plane proposal was a functionally passive element that implemented decisions taken by a central network controller. To enable the node-local control layer, the node controller has become a decision-making element. The network segment controller can reserve resources network-wide. Nonetheless, transmission time slots not reserved by the segment controller can be managed locally at the node level, using the node-local control mechanism (Fig. 3). This mechanism leverages he high-speed control channel to enable a signaling system that informs, per time slot and per wavelength (λ), whether the time slot is reserved by the network controller and, if not, whether it has been used by another node to transmit opportunistically. The node also monitors client data buffers to assess if there are frames to be inserted. These flags and information are processed in few tens of nanoseconds since the decision on the λ and flow insertion at the incoming time slot needs to be made in negligible time with respect to the time slot (< 1.5 μs).

Real-time node-local control enables two features relevant for optical slot switched networks: first, opportunistic traffic insertion to use idle optical transmission resources (unused time slots) and, second, the insertion of clock-maintaining optical slots when a wavelength to be dropped in the next node is carrying "empty time slots". The latter allows to avoid loss of data at the receiver due to Clock and Data Recovery (CDR) constraints, which after relatively long periods (few μs) without receiving optical data can lose track of the data reference clock. Fig. 3, shows the node-local control mechanism in action. In this case, the third time slot of the 10-slot schedule window is not reserved and is left for opportunistic traffic insertion. In the first reservation window (left side), opportunistic traffic insertion mechanism was used. Local controller of Slave 1 detected opportunistic data stored in transmission buffer (local monitoring) and an un-used and non-reserved slot was in transit (control channel flags). Thus, the third slot (indicated with the letter "c" in the figure) was used for opportunistic transmission. Slave 2 (S2) detects that this slot was used by a previous node (S1) and did not make any insertion of this λ during this time slot (letter "a" in the figure). In the following reservation window, right side of Fig. 3, S1 had no information at opportunistic queues, so the slot remains empty ("a"). Nevertheless, S2 detects that the time slot in the λ to be dropped at the next node is empty and inserts a clock-maintaining optical packet ("b") even if it did not have client information to send opportunistically. The lower part of Fig. 3 shows the reception at Master node (M) from both Slaves using node-local control. No empty nor overlapped slots are observed at reception. If the clock-maintaining optical packet was not inserted, the receiver would have experienced absence of signal when there is no opportunistic data inserted (e.g., time slot "b"). Note that the central controller is not aware of these local decisions. The decision on the insertion (λ and queue) is taken in 3 clock cycles (19.2 ns in this case). Node-local control enables the support of best-effort traffic (non-controlled opportunistic insertion) as well as time-sensitive traffic (controlled opportunistic insertion).



Slot #	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
Flags in slave 1 for lambda to master																				
Data in opp. Buffer	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
Reserved slot	1	1	0	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1
Used slot	0	0	~~0~~→1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Flags in slave 2 for lambda to master																				
Data in opp. Buffer	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reserved slot	1	1	0	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1
Used slot	0	0	→1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

a: empty optical slot b: clock-maintaining optical packet c: opportunistic optical packet

Fig. 3: node local control usage flags transmitted through the control channel used to take locally the decision on the wavelength and queue insertion at each time slot.

D. Real-time network testing device (NTD)

For increased precision and to perform experiments spanning both control and data plane, we have devised a real-time network-testing device (NTD). It is integrated into the same FPGA-board as the network controller, triggering data plane probe flows from control instructions to evaluate their impact on network performance. The NTD is composed by modular and independent flow generators that are activated independently, with few nanoseconds precision counters to perform per-flow and per-frame latency, jitter, and packet loss monitoring. It provides several high-speed (over 10 Gb/s) client interfaces. Given our requirements (time precision, reconfigurability, and evaluations spanning network's control plane and data plane), the integrated NTD outperforms commercially available devices.

4. Experimental evaluation

First, we evaluate the reconfiguration delay for a single network segment and assess the impact of the real-time segment controller on network dynamicity. Second, we evaluate the connection establishment time for multi-network segment environment, requiring the intervention of the global controller to establish end-to-end communication. Third, we implement real-time distributed processing in the intra-data center network domain to assess the impact of the utilization of node-local control scheme on the performance of this kind of application.

A. Network segment reconfiguration

For these experiments, we use the intra-data center network segment (3-nodes). Three experiments requiring real-time segment network controller were performed in this section: first the connection establishment time of a flow, with the request coming from the controller itself, requiring the segment controller to deliver the instructions to the corresponding node. Second, the real-time monitoring system was used, so the network segment controller could decide based on the current state of the network, see fig. 4. Third, a flow endpoint is modified, and the network automatically detects the change and reallocates resources to migrate the flow.

1. *Centralized connection establishment*

We evaluate the delay incurred by the network to establish a connection from the moment the segment controller retrieves a request. The controller processes the request, takes a decision, and sends the new transmission schedule to the alluded nodes. The connection establishment time (CET) accounts from the moment the request (and first data frame) are sent until the first data frame is retrieved at the destination node. We evaluate the worst case, where propagation spans three nodes (one for the instruction and two for data). Table 1 shows the CET breakdown. Note that networkwide transmission schedule update is applied. The overall CET is below 20 µs, from 2 ms when using non-real time control plane, reported in [4]. The main delay on the network-wide CET comes from propagation (>80%).

Table 1. Breakdown of connection establishment time

Overall CET	Controller /Master Tx	Propagation (control + data)	Node schedule update	Execution at node
< 20 µs	0.5 µs	16.5 µs	18.2 ns	< 1.5µs

2. *Automatic connection establishment*

In this experiment, we evaluate the delay incurred to establish a connection using the network monitoring function to decide on resource allocation. When the network controller detects frames on a given transmission buffer, it schedules optical slots so the information can be transmitted to the destination node. Fig. 4 shows the scheme of control and data plane contributions to the overall delay when considering communication from Slave 1 (S1) to Master (M). This is the worst CET case when applying automatic resource allocation, since monitoring information from S1 is propagated through two optical fiber segments to reach the controller (connected to M) and the data flow is also propagated through two fiber segments (S1→ S2, S2→ M) before reaching the destination node.

Latency breakdown with delays incurred by the control and data plane is shown in Fig. 5. The monitoring delay includes the time from the moment the first frame is retrieved at S1 client interface until the information of this event reaches the central controller. The decision time is defined as the time taken to evaluate the monitoring information and fetch in a look-up table (LUT) the transmission schedule to be implemented accordingly. In this case, the LUT allocates periodic time slots for the incoming flow to reach M. No schedule calculation is applied although the infrastructure allows for the scheduler application to run on the segment network controller. Instruction delay is the time taken to effectively deliver the new transmission schedule from the network segment controller to the source node (S1). Insertion delay is the time taken for the flow to reach the first time slot allowed for its transmission, while data transmission delay is the overall time taken for the frame to reach the destination client interface once transmission is allowed. It is important to note that since our system provides bounded and known control and data plane delay, we can guarantee a bounded CET. The target latency is guaranteed by providing enough margin to cover for both constant and variable delays. Per flow constant latency is enforced at the reception node by the jitter compensation mechanism (JCM) at reception as in [4]. The JCM is based on the systematic and precise time stamping of each frame at the insertion node and calculation of the experienced latency at the egress node. The difference between the experienced and target latency is solved through buffering before sending the frame to the destination client interface.

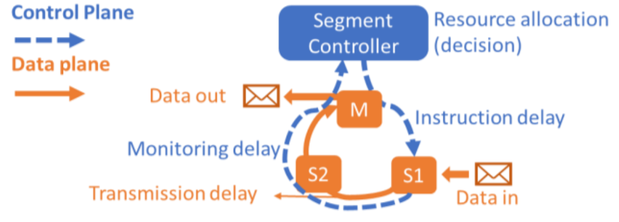


Fig. 4: Network segment controller experimental scheme

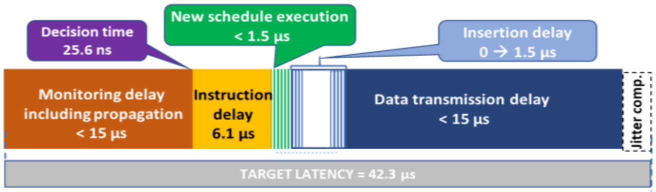


Fig. 5: Reconfiguration breakdown: control and data plane using Slave1 -Master communication as an example for control and data delays

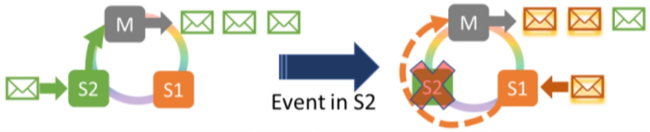
3. *Automatic flow reallocation*

Fig. 6 shows a use case of the real-time control plane in DDN: a time-sensitive flow with a predefined target latency is running and transmitting from a server located at S2 node to M node. The source endpoint is modified from S2 to S1, emulating a virtual machine migration within the data center using the real-time NTD. The network controller automatically detects the endpoint substitution through real-time monitoring and reconfigures the network accordingly (reallocating reservation from S2→M to S1→M). Since the pre-defined target latency of the flow is shorter than the complete reconfiguration time of the network, we can guarantee a hitless source endpoint modification from the destination client perspective. The flow is seamlessly retrieved even during the reconfiguration. Fig. 6, shows the latency experienced by the frames retrieved at M node. The endpoint substitution was completely transparent for the receiving endpoint, see flow's constant average latency (42.3µs) notwithstanding different source node. This was achieved thanks to the real-time control plane, that detects changing endpoints and reacts promptly, and to the JCM that identifies the latency experienced per frame. The JCM also detects the change of source node and inserts latency offset to account for the propagation difference, maintaining the service latency unchanged. In DDN context, we aim at guaranteeing performance in all scenarios, even when communication network goes through reconfiguration. We have shown that real-time network segment control is a useful tool enabling highly dynamic networks.

B. Multi-network segment reconfiguration

A real-time global controller has been implemented and evaluated for the first time, using the two network segments described in Section 3: intra- and inter-data center, see Fig. 7. It allows to treat flow requests spanning different network segments, establishing end-to-end communication.

We experimentally measured the delay incurred for multi-segment flow establishment. The real-time network testing device is used to send the request for the establishment of the communication and to start the



Flow	Latency (Avg.)	Jitter	Monitoring delay	Execution time
S1 to M	42.3 µs	12 ns	< 15 µs	< 9 µs
S2 to M	42.3 µs	12 ns	< 9 µs	< 15 µs

Fig. 6: intra data center network resource reallocation experiment. Flow characteristics remain unchanged even after source endpoint

data plane flows at the same time. Requests are sent to the global controller using IP packets on 10G Ethernet interfaces, containing the source and destination clients, required capacity and target latency.

The delay added by the FPGA-based global control layer infrastructure is shown in Table 2. The overall transmission time of the request frame, from source interface until reaching the global controller (*Request Tx*) is less than 1µs. After retrieving this request, the global controller defines the instructions to be sent to both network segments to establish the connection (Global *instruction generation*). It takes around 10 clock ticks in FPGA processing for the generation of the instructions, note that in our current implementation the clock period is 6.4 ns. The communication between global controller and network-segment controllers also takes less than 1 µs (*Instruction Tx*).

Table 2: real-time global controller added delay

Request Tx	Instruction generation	Instruction Tx
750 ns	< 70 ns	750 ns

1. *Multi-segment flow connection establishment time (CET)*

These experiments evaluate the end-to-end dynamicity of the network through the CET, which includes flow request delivery, processing at global controller, delivery of instruction to both network segment controllers, network-wide resource allocation and data frame end-to-end transmission. The requested flow starts at a server located at S1 in the edge-cloud data center (intra-DC) network and ends at the far end inter-DC node (M inter-DC node), this is the scenario with the largest propagation delay among all possible source/destination node pairs. Propagation distance were varied in the inter-DC network to evaluate the impact on end-to-end network dynamics, Link A, which connects both nodes (control and data plane), and Link B, connecting the global network controller and the segment controller, see Fig. 7.

Experiments A0, B0, C0 in Table 3 are benchmark cases for different Inter-DC segment lengths (Link A in Fig. 7), where resources were pre-allocated before the flow starts, thus, no network reconfiguration was required. This is the minimum time required for communication (including transmission, propagation and interfacing). The next experiments evaluate the reconfiguration time of both network segments independently and connected (end-to-end). We distinguished two regimes. In the first one (experiments A1, A2 and B1), the second network segment (inter-DC) reconfigures much faster than the first segment (intra-DC) since the inter-DC segment control plane delay is short in comparison with reconfiguration and data transmission delay of the intra-DC segment. In this regime, the second network segment was already reconfigured when it received information from the first segment, thus, its reconfiguration delay was eclipsed, and the flow experienced no additional delay due to reconfiguration of the inter-DC network, only the minimum data plane propagation delay (as in A0 and B0). On the contrary, in Experiments A3, B2, B3, C1, C2 and C3; the second network segment becomes slower than the first due to added propagation between inter-DC nodes and between the global controller and inter-DC network controller. In these cases, the first network was reconfigured and the information was delivered to the gateway node of the second network even before it was configured, thus, the reconfiguration and data transmission delay incurred by the first network were transparent for the overall connection establishment time (CET), which, as a result, is the same as if there was only the inter-DC network segment.

Table 3: multi-segment connection establishment time

Experiment	A0	A1	A2	A3	B0	B1	B2	B3	C0	C1	C2	C3
Link A [km]	**0.03**	0.03	0.03	0.03	**1.6**	1.6	1.6	1.6	**5.75**	5.75	5.75	5.75
Link B [km]	N/A	0.03	3.2	11.7	N/A	0.03	3.2	11.7	N/A	0.03	3.2	11.7
Inter-DC CET [µs]	2.1	6.8	23.1	**64.1 ***	10.0	22.7	**38.0 ***	**79.4 ***	29.7	**62.4 ***	**78.6 ***	**120.3 ***
Intra-DC CET [µs]	14.6	**24.2 ***	**24.2 ***	24.2	14.6	**24.2 ***	24.2	24.2	14.6	24.2	24.2	24.2
End-to-end CET [µs]	**16.1**	**25.6**	**25.6**	**64.1**	**23.8**	**33.7**	**38.0**	**79.5**	**43.2**	**62.5**	**78.6**	**120.3**

*** dominant network segment**

Given the reconfiguration delay $D_{reconf\,i}$, the data propagation delay in segment $D_{data.\,i}$, with $i \in$ {intra-DC = 1, inter- DC = 2} the CET follows the following expression:

$$CET = \mathbf{max}\{\, D_{reconf\ 1} + D_{data\ 1}\ , D_{reconf\ 2}\} \ + D_{data\ 2} \qquad (1)$$

Fig. 8 shows the CET evolution, the abscissa represents Link B length and the ordinate the Link A length. Equation (1) defines the dashed line in Fig. 8 when $D_{reconf\,2} = D_{reconf\,1} + D_{data\,2}$. Below the dashed line, the reconfiguration time of the inter-DC segment is eclipsed by the reconfiguration and data transmission of intra-DC segment as can be inferred from (1). In this region, we can observe that two dots having the same Link A length, share the same CET (25.6 µs), since link B only affects inter-DC control plane delay ($D_{Reconf\,2}$) which is eclipsed in this region. Link A in the other hand, transports data and control channel in the inter-DC network, affecting both planes delay ($D_{reconf2}$ and $D_{data\,2}$), so even if control instruction propagation is eclipsed, data propagation in the second network segment affects the overall CET. All dots above the dashed line are cases where reconfiguration time of the second network segment (inter-DC) is larger than reconfiguration and data transmission

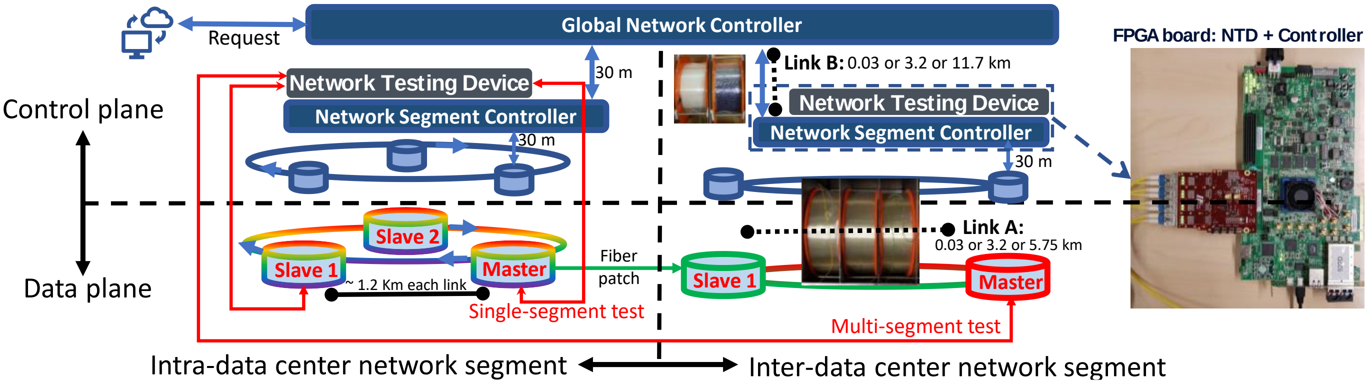

Request
Global Network Controller
Control plane
Data plane
Network Testing Device
Network Segment Controller
30 m
30 m
Slave 2
Slave 1
Master
~ 1.2 Km each link
Fiber patch
Single-segment test
Link B: 0.03 or 3.2 or 11.7 km
Network Testing Device
Network Segment Controller
30 m
Link A: 0.03 or 3.2 or 5.75 km
Slave 1
Master
Multi-segment test
FPGA board: NTD + Controller
Intra-data center network segment
Inter-data center network segment

Fig. 7: general experimental scheme for single- and multi-segment evaluation. Global control layer managing an intra-data center (intra-DC) and inter-data center (inter-DC). Variable length links are shown in dotted black lines (Link A and Link B).

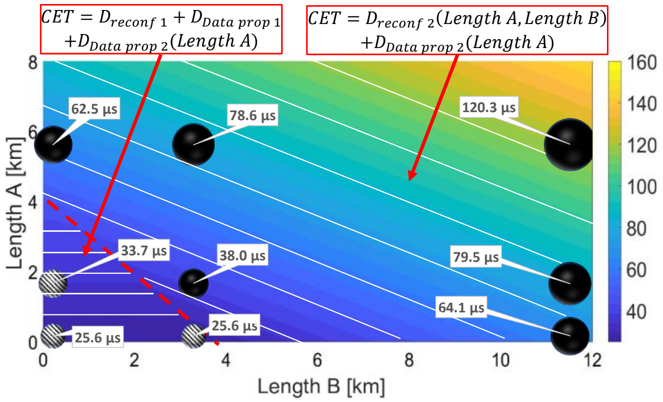


Fig.8: Multi-segment measured connection establishment time in µs (dot size) when varying Links A and B lengths. The color map represents the theoretical CET variation as a function of A and B lengths.

of the first network segment, thus, the communication establishment time is the same as if only the second network segment was there.

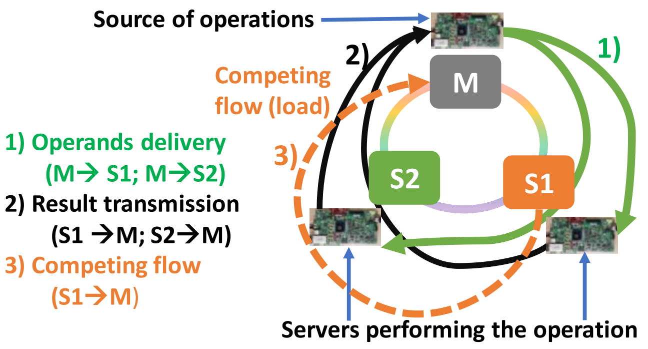
We can conclude from (1) that optimizations on control and data plane delay do not necessarily improve multi-segment CET. It is required first to identify which network segment is dominant (slower) and optimize the delays contributing to the CET.

In general terms, the reconfiguration time of two network segments is shorter than that of the addition of the reconfiguration time of each network segment separately because the reconfiguration time and data delivery in the first segment is discounted from that of the second network reconfiguration time. The value of resource pre-allocation or predictive network control became evident to reduce CET when possible, as shown by experiments A0, B0, C0.

C. Application case: distributed computing

As described in the first section, different application can find high value on time performance guarantees and high dynamics. This is conspicuous for applications such as high-frequency trading where additional milliseconds of delay could represent even hundreds of million dollars for big brokerage firms [28], these firms could leverage our solution to deploy computing capacity taking trading decisions in the edge DDN data center as close as possible to the execution venue for reduced latency and having the ability to adapt their computing capacity in real-time; remote or computer-assisted surgery requires also a bounded latency and guaranteed communication to provide high-quality service in a context where human health and life are at risk [29]; mobile multiplayer gaming, where latency must be respected and the network should be dynamic enough to adapt as the end user moves while maintaining a continuous and acceptable user experience [30]; finally, Industry 4.0, a major 5G driver, where robots and devices located in the factory floor can use mobile access to establish communication with the cloud to exchange mission-critical control information [31, 32]. These use cases fit into our application scenario where the processing capacity is located (and potentially distributed) in the edge-cloud. In these scenarios, fast network reconfiguration might be required due to the mobility of one endpoint, as well as the need of additional processing capacity or the migration of virtual machines in the edge-cloud.

In previous experimental work [4], we focused on the data plane determinism and proved that we can provide constant latency within the network notwithstanding the network utilization or the flow throughput, given that sufficient capacity (periodic time slots) is reserved for the deterministic flows. In this work, we use the same deterministic infrastructure, but we focus on providing high dynamics measured as the network reconfiguration time by implementing the real-time control plane and assess the impact of this dynamicity on distributed applications relying on the network.



Source of operations
1)
2)
Competing flow (load)
M
3)
S2
S1
1) Operands delivery (M→ S1; M→S2)
2) Result transmission (S1 →M; S2→M)
3) Competing flow (S1→M)
Servers performing the operation

Fig. 9: Experimental scheme for distributed operations. 2000 distributed operations requested by a server located at Master node to servers at Slaves nodes in the intra-data center network segment.

To compare different network control approaches and assess the value of real-time control plane, we emulated real-time distributed computing over the intra-data center network: operands are sent from an FPGA-based client at Master node to FPGA servers connected to client interfaces at Slave nodes, see Fig. 9.

Once the server at Master retrieves operation results from Slaves, it sends the next operands to be processed. These experiments evaluate the latency of the first-received packet for all established flows. This application is highly sensitive to reconfiguration time since communication resources need to be allocated each time results need to be transmitted. A flow competing for transmission resource to communicate with Master node was added to stress the network and study its incidence on the performance of the distributed application.

We evaluated the completion time of 2000 operations under different control approaches:

1. *Network segment real-time control*

In this approach, the communication between Master node and Slaves was pre-allocated while the communication of the results from Slaves to Master node had to be dynamically allocated and released by the network segment controller. The network controller monitors the filling of transmission queues at Slaves, when buffered information is detected, the controller sends the instruction to schedule the transmission. Fig. 10 shows completion time for different scenarios. The blue line at the bottom is the benchmark, showing the completion time for 2000 operations when all required transmissions are pre-established. Thus, this is the lower bound of the completion time (accounting for transmission, propagation and interfacing).

All other cases shown in Fig. 10 required systematic delivery of monitoring data from nodes, centralized decision and instruction distribution to establish communication.

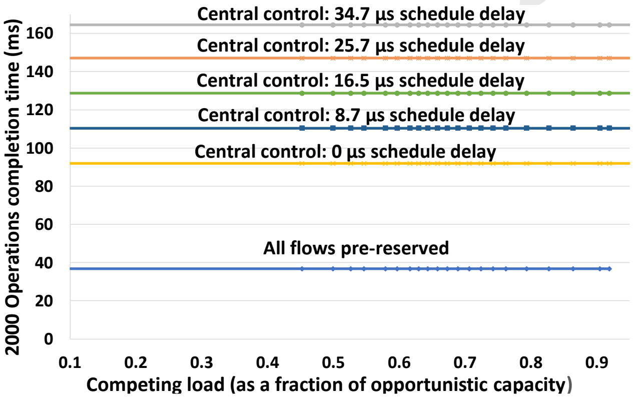


Fig. 10: Completion time for 2000 distributed operations using central network segment controller for resource allocation

We emulated the schedule processing time through a fixed delay before the controller delivers new resource allocation (from 0 to 34.7 µs). The yellow line (second from the bottom), shows the completion time lower bound that can be provided when resource allocation (monitoring + instruction transmission) is done centrally by the real-time segment controller, and no schedule processing delay exists. We can observe the incidence of the schedule computation delay on the overall network performance on the following lines: task completion time increases with schedule processing time. Note that the performance (completion time) is not affected by the competing flow load since the transmission resources are reserved for the results to be sent.

2. Node local control with no fairness mechanism

To reduce the delay induced by networkwide transmission resource reservation, we used opportunistic slot allocation enabled by the node-local control. In the experiment, only 1/10 un-reserved slot was left to be used opportunistically (opportunistic capacity) shared by both Slaves to communicate results, while 9/10 of network transmission resources were reserved for other deterministic flows.

Fig. 11 shows in dashed line the completion time when using opportunistic slot allocation. This control approach enables best effort traffic insertion, which is a challenge for time-slotted optical networks. For low competing flow load, sheer opportunistic resource allocation outperforms real-time central segment control since the decision on the insertion is taken locally at the node, it allows to insert frames containing results as soon as an unused slot is available (which is likely in low load scenario).

In very low load, opportunistic frame insertion performance is comparable to that of the pre-allocated scenario. However, when there are competing opportunistic flows using an important fraction of opportunistic resources, and no starvation control or fairness mechanism is applied, the system takes longer to complete the operations. Competing flow load increases opportunistic slots that are taken by the competing flow, leaving fewer opportunities for the intermediate node to transmit results. This dependency on network load is not suited for Deterministic and Dynamic Networks since we aim at providing performance guarantees notwithstanding network's usage or conditions.

3. Node local control with basic fairness mechanism

To solve the unbounded performance featured by sheer opportunistic transmission approach, a simple fairness mechanism was applied to avoid starvation of intermediate nodes. It consisted on limiting to 1 slot every 2 (or 4) available opportunistic slots to be used by any opportunistic flow. Fig. 12 shows in dashed lines the results of using this mechanism. Completion time is kept constant from moderate until high competing flow load, since we guarantee the availability of opportunistic slots to intermediate nodes.

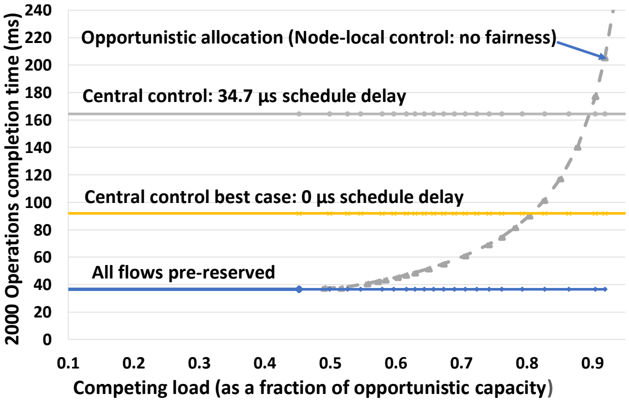


Fig. 11: Completion time for 2000 distributed operations using node-local control without fairness or flow control

This simple approach allows to guarantee bounded and low latency when using node-local control approach. In *DDN* context this is critical since we show that node local control mechanism not only supports best effort traffic but can also be used for time-sensitive traffic.

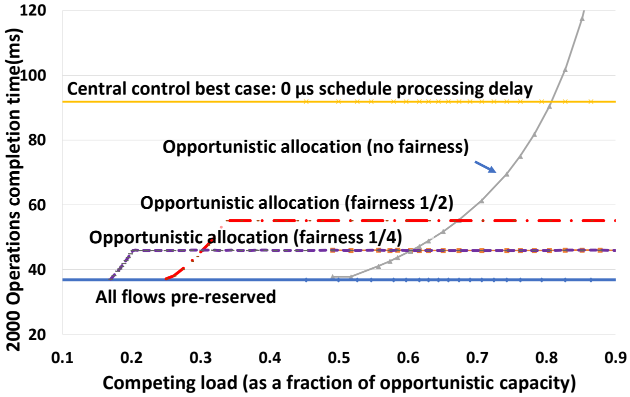


Fig. 12: Completion time for 2000 distributed operations using node-local control with starvation avoidance

5. Conclusions

Real time control plane proved to be a key value for deterministic and dynamic networks in next-generation edge-cloud environments. We proved that the infrastructure enabling end-to-end determinism in optical networks can also be highly dynamic by applying real-time control plane strategies at all control layers. In the case of multi-segment networks, we proved for the first time in DDN context that end-to-end multi-segment service establishment time can be in the order of hundreds of µs in edge-cloud environment with the real-time global control layer in place. We observed that multi-segment connection establishment time enhancement requires identifying the dominant network segment and reducing its data propagation and/or reconfiguration delays. In single segment networks we showed how real-time network segment control allowed to monitor the state of networking elements, take decisions and deploy resources in tens of µs. Real-time network segment control enables transparent migration of communication endpoints while guaranteeing constant latency per flow when target latency is greater than the propagation and reconfiguration time of the network (~10 µs). When maximum dynamicity is required, real-time node-local control is the suitable control solution, allowing for control decisions in tens of nanoseconds, enabling opportunistic traffic insertion. Node-local control also enables the utilization of idle resource by non-time sensitive best-effort flows, allowing to increase the overall network efficiency. Latency determinism and ultra-fast control do not come for free. To enforce constant latency, we perform delay equalizing buffering at reception, thus requiring additional memory resources for this purpose. We achieve ultra-fast dynamics through dedicated and high-speed control channel and FPGA-based controllers, which can be costly. We count first on the assumption that deterministic flows represent a small fraction of the overall capacity and the infrastructure can be dimensioned accordingly. Second, in the edge-cloud network context, we envision to have small-sized edge data centers connecting to few access segments, thus, the resource scheduling decisions are made over a reduced number of network nodes, decreasing the processing complexity at the network controller. The assessment of the relative cost and benefits of DDN is a field to be explored in future work.

REFERENCES

[1] Y. Pointurier, N. Benzaoui, W. Lautenschlaeger and L. Dembeck, "End-to-End Time-Sensitive Optical Networking: Challenges and Solutions," *Journal of Lightwave Technology,* vol. 37, no. 7, pp. 1732-1741, 2019.

[2] W. A. Khan, L. Wisniewski, D. Lang and J. Jasperneite, "Analysis of the requirements for offering industrie 4.0 applications as a cloud service," in *2017 IEEE 26th International Symposium on Industrial Electronics (ISIE)*, Edinburgh, 2017.

[3] A. Nasrallah, A. Thyagaturu, Z. Alharbi, C. Wang, X. Shao, M. Reisslein and H. ElBakoury, "Ultra-Low Latency (ULL) Networks: The IEEE TSN and IETF DetNet Standards and Related 5G ULL Research," *IEEE Communications Surveys & Tutorials,* vol. 21, no. 1, pp. 88-145, 2019.

[4] N. Benzaoui, M. Szczerban, J. M. Estarán, H. Mardoyan, W. Lautenschlaeger, U. Gebhard, L. Dembeck, S. Bigo and Y. Pointurier, "Deterministic Dynamic Networks (DDN)," *Deterministic Dynamic Networks (DDN),* pp. 365-3474, 2019.

[5] IEEE, "IEEE 802.1 Time-Sensitive Networking Task Group," [Online]. Available: https://1.ieee802.org/tsn/. [Accessed 27 02 2020].

[6] D. Saha, B. Rajagopalan and G. Bernstein, "The optical network control plane: state of the standards and deployment," *IEEE Communications Magazine,* pp. S29-S34, 2009.

[7] ITU-T Recommendation G.8080, *Architecture for the automatically switched optical network (ASON),* 2006.

[8] IETF RFC 3471, *GMPLS Signaling Functional Description,* 2003.

[9] L. Y. Ong, E. Roch, S. Shew and A. Smith, "New Technologies and Directions for the Optical Control Plane," vol. 30, no. 4, pp. 537-546, 2012.

[10] IETF RFC 7426, *Software-Defined Networking (SDN): Layers and Architecture Terminology,* 2015.

[11] ITU-T Recommendation G.7702, *Architecture for SDN control of transport networks,* 2018.

[12] Open Networking Foundation, *SDN Architecture,* 2014.

[13] H. Yang, J. Zhang, Y. Zhao, Y. Ji, H. Li, Y. Lin, G. Li, J. Han, Y. Lee and T. Ma, "Performance evaluation of time-aware enhanced software defined networking (TeSDN) for elastic data center optical interconnection," *Optics Express,* vol. 22, no. 15, pp. 17630-17643, 2014.

[14] G. M. Saridis, S. Peng, Y. Yan, A. Aguado, B. Guo, M. Arslan, C. Jackson, W. Miao, N. Calabretta, F. Agraz, S. Spadaro, G. Bernini, N. Ciulli, G. Zervas, R. Nejabati and D. Simeonidou, "Lightness: A Function-Virtualizable Software Defined Data Center Network With All-Optical Circuit/Packet Switching," *Journal of Lightwave Technology,* vol. 34, no. 7, pp. 1618-1627, 2016.

[15] K. Kondepu, C. Jackson, Y. Ou, A. Beldachi, A. Pagès, F. Agraz, F. Moscatelli, V. K. N. C. W. Miao, G. Landi, S. Spadaro, S. Yan, D. Simeonidou and R. Nejabati, "Fully SDN-enabled all-optical architecture for data center virtualization with time and space multiplexing," *IEEE/OSA Journal of Optical Communications and Networking,* vol. 10, no. 7, pp. 90-101, 2018.

[16] M. Yuang, P.-L. Tien, W.-Z. Ruan, T.-C. Lin, S.-C. Wen, 3. C.-C. L. Po-Jen Tseng, C.-N. Chen, C.-T. Chen, Y.-A. Luo, M.-R. Tsai and S. Zhong, "OPTUNS: Optical intra-data center network architecture and prototype testbed for a 5G edge cloud," *IEEE/OSA Journal of Optical Communications and Networking,* vol. 12, no. 1, pp. A28-A37, 2020.

[17] F. Yan, W. Miao, O. Raz and N. Calabretta, " Opsquare: A flat DCN architecture based on flow-controlled optical packet switches," *IEEE/OSA Journal of Optical Communications and Networking,* vol. 9, no. 4, pp. 291-303, 2017.

[18] X. Xue, "Experimental Assessment of SDN-Enabled Reconfigurable OPSquare Data Center Networks with QoS Guarantees," in *2019 Optical Fiber Communications Conference and Exhibition (OFC)*, San Diego, CA, USA, 2019.

[19] M. Boehm, J. Ohms, M. Kumar, O. Gebauer and D. Wermser., "Time-Sensitive Software-Defined Networking: A Unified Control- Plane for TSN and SDN," in *Mobile Communication - Technologies and Applications; 24. ITG-Symposium*, Osnabrueck, Germany, 2019.

[20] E. Chung, J. Fowers, K. Ovtcharov, M. Papamichael, A. Caulfield, T. Massengill, M. Liu, M. Ghandi, L. D, S. Reinhardt, S. Alkalay, H. Angepat, D. Chiou, A. Forin, D. Burger, L. Woods, G. Weisz, M. Haselman and D. Zhang, "Serving DNNs in Real Time at Datacenter Scale with Project Brainwave," *IEEE Micro,* vol. 38, no. 2, pp. 8-20, 2018.

[21] M. Szczerban, J. M. Estarán, N. Benzaoui, H. Mardoyan, Y. Pointurier and S. Bigo, " Real-time control for deterministic and dynamic networks," in *45th European Conference on Optical Communication (ECOC 2019)* , Dublin, Ireland, 2019.

[22] M. Szczerban, J. Estarán, N. Benzaoui, H. Mardoyan and Y. Pointurier, "Real-Time Node Local Control for Ultra-Dynamic and Deterministic All-Optical Intra Data Center Networks," in *2020 Optical Fiber Communications Conference and Exhibition (OFC)*, San Diego, CA, USA, 2020.

[23] ITU-T Recommendation G.7701, *Common control aspects,* 2016.

[24] T. Verolet, A. Gallet, X. Pommarède, J. Decobert, D. Make, J. Provost, M. Fournier, C. Jany, S. Olivier, A. Shen and G. Duan, "Hybrid III-V on Silicon Fast and Widely Tunable Laser Based on Rings Resonators with PIN Junctions," in *2018 Asia Communications and Photonics Conference (ACP)*, Hangzhou, 2018.

[25] R. C. Figueiredo, N. S. Ribeiro, A. M. O. Ribeiro, C. M. Gallep and E. Conforti, "Hundred-Picoseconds Electro-Optical Switching With Semiconductor Optical Amplifiers Using Multi-Impulse Step Injection Current," *Journal of Lightwave Technology,* vol. 33, no. 1, pp. 69-77, 2015.

[26] P. Papaphilippou, J. Meng and W. Luk, "High-Performance FPGA Network Switch Architecture," in *The 2020 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, Seaside, CA, USA.

[27] N. Benzaoui, "CBOSS: bringing traffic engineering inside data center networks," *IEEE/OSA Journal of Optical Communications and Networking,* vol. 10, no. 7, pp. 117-125, 2018.

[28] C. Moallemi and M. Sağlam, "The Cost of Latency in High-Frequency Trading," *Operations Research,* vol. 61, no. 5, pp. 1069-1257, 2013.

[29] J. Marescaux, J. Leroy, F. Rubino, M. Smith, M. Vix and M. S. D. Mutter, "Transcontinental Robot-Assisted Remote Telesurgery: Feasibility and Potential Applications," *Annals of Surgery,* vol. 235, no. 4, pp. 487-492, 2002.

[30] A. I. Wang, M. Jarrett and E. Sorteberg, "Experiences from Implementing a Mobile Multiplayer Real-Time," *International Journal of Computer Games Technology,* vol. 2009, 2009.

[31] W. A. Khan, L. Wisniewski, D. Lang and J. Jasperneite, "Analysis of the requirements for offering industrie 4.0 applications as a cloud service," in *2017 IEEE 26th International Symposium on Industrial Electronics*, Edinburgh, 2017.

[32] M. Wollschlaeger, T. Sauter and J. Jasperneite, "The Future of Industrial Communication: Automation Networks in the Era of the Internet of Things and Industry 4.0," *IEEE Industrial Electronics Magazine,* vol. 11, no. 1, pp. 17-27, March 2017.