# Building Network Nervous System with Multilayer Telemetry to Realize AI-assisted Reflexes in Software-defined IP-over-EONs for Application-aware Service Provisioning

**Hongqiang Fang, Wei Lu, Lipei Liang, Bingxin Kong, Zuqing Zhu**
*University of Science and Technology of China, Hefei, Anhui 230027, China, Email: zqzhu@ieee.org*

**Abstract:** We design and experimentally demonstrate a network nervous system that can leverage multilayer telemetry to realize artificial intelligence (AI) assisted network reconfigurations (*i.e.*, reflexes) in a software-defined IP over elastic optical network, for application-aware provisioning.

**OCIS codes:** (060.1155) All-optical networks; (060.4261) Networks, protection and restoration.

## 1. Introduction

Recently, IP over elastic optical networks (IP-over-EONs) have attracted intensive interests since they can effectively combine the benefits of IP and EON layers to handle the ever-growing network applications that have various quality-of-service (QoS) requirements [1]. Meanwhile, the programmability and application-awareness of IP-over-EONs can be further enhanced by leveraging the centralized network control and management (NC&M) provided by software-defined networking (SDN), *i.e.*, realizing software-defined IP-over-EONs (SD-IPoEONs) [2]. However, despite of the advantages, it is never easy to comprehensively monitor a complex network system as SD-IPoEON and realize timely and fine-grained network adjustments in it, for addressing the specific need of each individual application. This is because application-level monitoring would flood the network controller with tremendous status data, and each application might define an unique way for determining exceptions and making network adjustments. On the other hand, the issue cannot be resolved solely from the application side either, since the applications usually only know end-to-end (E2E) QoS parameters, but are not aware of what happening inside the network.

This dilemma motivates us to consider leveraging the joint effort of both the network controller and the applications. Specifically, the applications realize application-level (App-level) monitoring in a distributed manner and only send alarms out when there are exceptions, while the controller collects the digested information (*i.e.*, the alarms) to combine with its own lightpath-level ($\lambda$-level) monitoring for reaching intelligent NC&M decisions. This mechanism actually mimics the principle of the nervous system of human body. Hence, we refer to it as a network nervous system (NNS), which consists of a few App-level monitors (*i.e.*, the sensors) and an artificial intelligence assisted (AI-assisted) network controller (*i.e.*, the brain). In this work, we lay out the design of the NNS for an SD-IPoEON, prototype it in a real network testbed, and demonstrate its effectiveness for application-aware service provisioning experimentally.

## 2. Network Nervous System

Fig. 1(a) shows our design of the NNS for an SD-IPoEON. The data plane is a common IP-over-EON, where the EON layer consists of fiber links and bandwidth-variable wavelength-selective switches (BV-WSS') for setting up lightpaths, and the IP layer is built with OpenFlow switches (OF-SWs) and application servers. The OF-SWs are equipped with optical ports for communicating with each other through the EON layer, and the servers run applications with various QoS requirements. The control plane consists of a centralized controller and a few distributed App-level monitors. Based on its QoS requirements, each App-level monitor collects concerned E2E QoS parameters (*e.g.*, latency

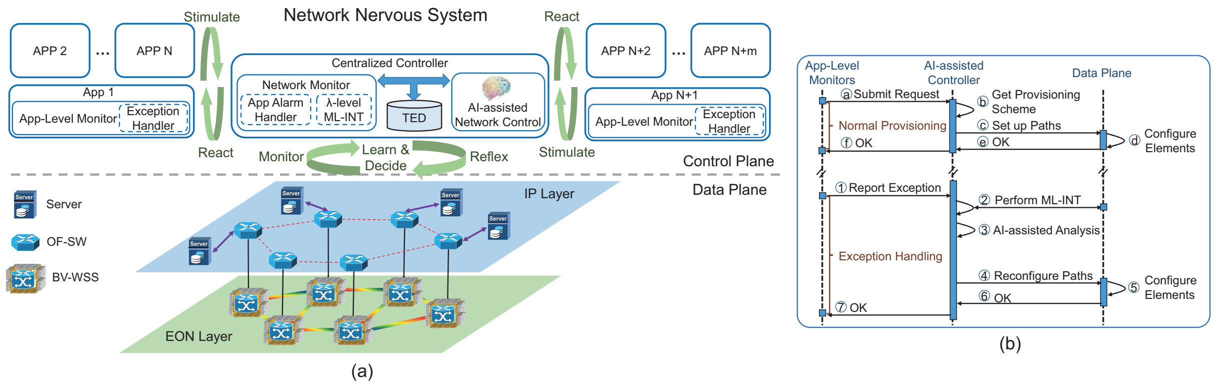


Fig. 1. (a) Architecture of NNS, APP: application, ML-INT: multilayer telemetry, TED: traffic engineering database, OF-SW: OpenFlow switch, BV-WSS: bandwidth-variable wavelength-selective switch, (b) Operation procedures of NNS.

and jitter) and uses a threshold-based mechanism to flag exceptions. Meanwhile, the AI-assisted controller performs multilayer telemetry (ML-INT) on each lightpath to gather $\lambda$-level monitoring results such as packet loss rate, optical signal-to-noise ratio (OSNR), *etc*, and combines them with the alarms from the App-level monitors as the input to its AI module. The AI-assisted network control module has been trained to learn the correlation among applications' multilayer provisioning schemes, their affected QoS parameters, the multilayer telemetry results, and hard/soft failure scenarios to identify and locate the root causes for the alarms accurately. Here, the multilayer provisioning schemes of the applications are obtained from the traffic engineering database (TED). Then, the controller invokes necessary network adjustments (*i.e.*, AI-assisted reflexes) to restore the affected QoS parameters. Fig. 1(b) illustrates the operation procedures for normal provisioning (*Steps a-f*) and exception handling (*Steps* 1-7).

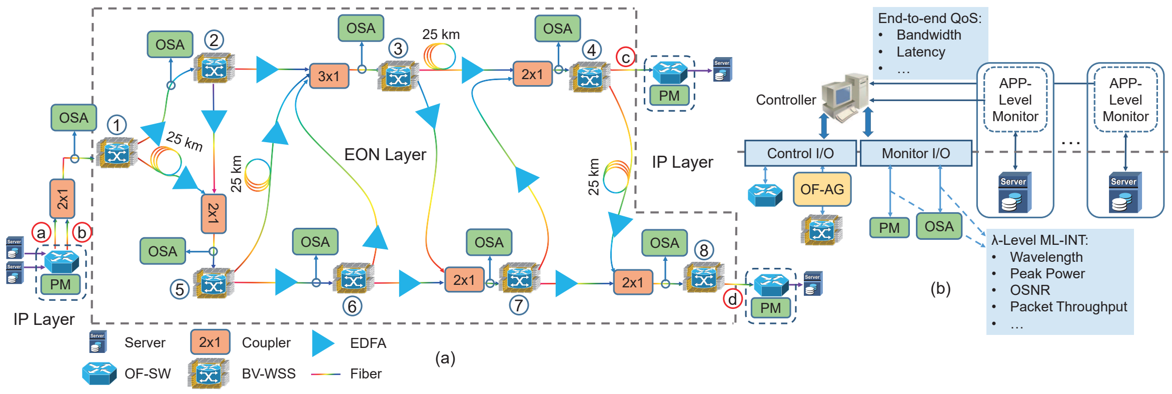


Fig. 2. SD-IPoEON testbed with NNS, (a) Data plane configuration, PM: port monitor, OSA: optical spectrum analyzer, EDFA: Erbium-doped fiber amplifier, and (b) Control plane configuration, OF-AG: OpenFlow agent.

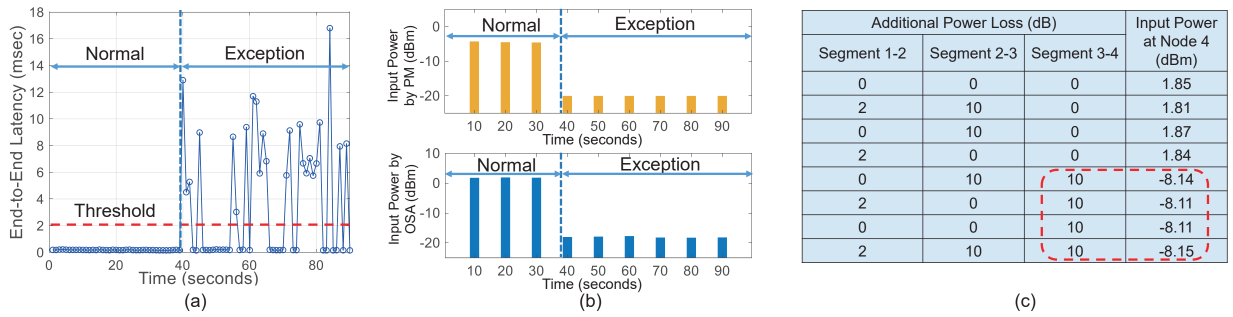
## 3. Experimental Testbed and Demonstrations

To verify the effectiveness of the proposed NNS, we build an SD-IPoEON testbed, prototype the NNS in it, and perform proof-of-concept demonstrations. The SD-IPoEON's data plane in Fig. 2(a) includes 8 optical nodes and 13 fiber links in the EON layer, and 3 OF-SWs and 4 servers in the IP layer. Here, each optical node is built with commercial $1\times9$ BV-WSS', and each OF-SW equips with one or more 10 Gbps optical transceivers. Specifically, the OF-SW that directly connects to *Node* 1 has two optical transceivers (*Ports a* and *b*) whose central wavelengths are 1551.72 and 1552.52 nm, respectively, while those connecting to *Nodes* 4 and 8 (*Ports c* and *d*) operate at 1548.52 and 1550.92 nm, respectively. To realize $\lambda$-level monitoring, we insert 8 optical spectrum analyzers (OSAs) in the EON layer, which run automatic scripts to collect and analyze optical spectra and report key metrics such as central wavelength(s), peak power and OSNR, and we also program a port monitor (PM) on each OF-SW to collect metrics related to its optical transceiver(s), *e.g.*, central wavelength, in/out optical power, and packet throughput. The collected metrics are sent to the AI-assisted controller in the control plane, as shown in Fig. 2(b). The controller is developed based on the ONOS platform [3] and TensorFlow, and it manages the BV-WSS' and the OF-SWs through OpenFlow agents (OF-AGs) and direct connections, respectively. We implement each App-level monitor based on iPerf [4], which runs in each server to gather the applications' E2E QoS parameters, including bandwidth, latency, packet loss rate, *etc*.

In the experiments, we set up a lightpath between *Ports b* and *c* using path 1-2-3-4, and use it to carry application traffic. Meanwhile, we turn on all the remaining electrical/optical devices in the testbed to emulate a relatively complicated network environment. The experiments consider two scenarios. In the first scenario, we assume that the application using the lightpath is delay-sensitive and its exception threshold on E2E latency is 2 msec. Then, with the E2E latency in Fig. 3(a), the App-level monitor detects exceptions from around $t = 40$ seconds. However, it is not a easy task to quickly locate the root cause of such exceptions in an SD-IPoEON. This is because the excessive E2E latency can be induced by a number of soft failures, such as congestion(s) in OF-SW(s), malfunctions of optical transceivers, excessive power loss on the lightpath, *etc*. Fortunately, our AI-assisted controller has been trained to get the job done quickly. Upon receiving the exception reports from the App-level monitor, the controller pulls out the $\lambda$-level monitoring results from the PM for *Port c* and the OSA connecting to the input of *Node* 4, since they are the closest to the lightpath's destination. Then, the controller finds that there are sudden drops on the input power measured by the PM and the OSA (as shown in Fig. 3(b)), which have time correlations with the E2E delay increases in Fig. 3(a)[1]. Hence, the exceptions are most likely caused by the excessive power loss on the lightpath. Next, based on

[1]The prolonged exceptions in Figs. 3(a) and 3(b) are shown for the sake of explaining the time correlation between the E2E latency and power loss, while in the actual operation of our system, exceptions can be detected and resorted within a few seconds.

its previous training with the samples in Fig. 3(c), the AI-assisted controller determines that the part that has excessive power loss should be segment 3-4, and decides to reroute the lightpath over path 1-2-5-6-7-4 and restore the E2E latency. The training samples in Fig. 3(c) basically teach the controller that for the concerned lightpath, excessive power drops at the input of *Node* 4 are strongly correlated with the additional loss on segment 3-4. This can be understood as that the erbium-doped fiber amplifier (EDFA) on segment 3-4 has relatively small gain and/or dynamic range.



| Additional Power Loss (dB) | | | Input Power at Node 4 (dBm) |
|---|---|---|---|
| Segment 1-2 | Segment 2-3 | Segment 3-4 | |
| 0 | 0 | 0 | 1.85 |
| 2 | 10 | 0 | 1.81 |
| 0 | 10 | 0 | 1.87 |
| 2 | 0 | 0 | 1.84 |
| 0 | 10 | 10 | -8.14 |
| 2 | 0 | 10 | -8.11 |
| 0 | 0 | 10 | -8.11 |
| 2 | 10 | 10 | -8.15 |

Fig. 3. Experimental results of exception scenario on E2E latency, (a) E2E latency from App-level monitor, (b) Power measurements from $\lambda$-level monitoring, and (c) Training samples for AI-assisted controller.

Note that, in the first scenario, the sudden power loss on the lightpath can also be detected by a threshold-based mechanism, and thus it cannot fully justify the necessity and advantage of the proposed NNS. Therefore, for the second scenario, we consider the case in which the exceptions are induced by the combined effect of multiple root causes, which can hardly be detected with a threshold-based mechanism. Here, we assume that the application using the lightpath is bandwidth-sensitive and cannot tolerate an E2E bandwidth lower than 8 Gbps. Fig. 4(a) shows the E2E bandwidth measured by the App-level monitor, which starts to flag exceptions to the controller since $t \approx 55$ seconds. Nevertheless, the related $\lambda$-level monitoring results in Fig. 4(b), which are the input power collected by the PM for *Port c* and the OSNR measured by the OSA connecting to the input of *Node* 4, do not show clear time correlation with the exceptions. Promisingly, based on the learned correlation among the input power, the OSNR, and the operation state of the lightpath on E2E bandwidth (*i.e.*, with the training samples in Fig. 4(c)), the controller can still quickly determine that the exceptions are caused by the combined effect of OSNR degradation (major) and power loss (minor) on the lightpath. Note that, the data points for normal operations and exceptions actually overlap with each other in Fig. 4(c), and thus we cannot detect the exceptions with a simple threshold-based mechanism. Next, the controller can investigate more to locate the soft failure on the lightpath and restore the applications's E2E bandwidth accordingly.

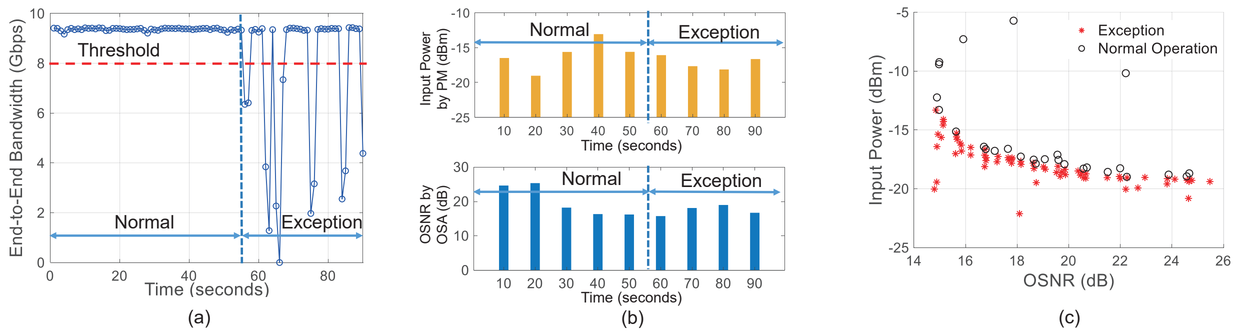


Fig. 4. Experimental results of exception scenario on E2E bandwidth, (a) E2E bandwidth from App-level monitor, (b) Input power and OSNR measured by $\lambda$-level monitoring, and (c) Training samples for AI-assisted controller.

## 4. Summary

We designed and experimentally demonstrated the NNS for realizing application-aware service provisioning in an SD-IPoEON. Experimental results indicated that with the NNS, our AI-assisted network controller can leverage multilayer telemetry results to detect and locate the root causes of application exceptions and resolve them successfully.

## References

[1] O. Gerstel *et al.*, "Multi-layer Capacity Planning for IP-Optical Networks," *IEEE Commun. Mag.*, vol. 52, pp. 44-51, Jan. 2014.
[2] S. Liu *et al.*, "On the Cross-Layer Orchestration to Address IP Router Outages with Cost-Efficient Multilayer Restoration in IP-over-EONs," *J. Opt. Commun. Netw.*, vol. 10, pp. A122-A132, Jan. 2018.
[3] ONOS. [Online]. Available: http://onosproject.org/.
[4] iPerf. [Online]. Available: https://iperf.fr.