Open-Source QoT Estimation for Impairment-Aware Path Computation in OpenROADM Compliant Network

Ahmed Triki[(1)], Esther Le Rouzic [(1)], Olivier Renais[(1)], Guillaume Lambert[(1)], Gilles Thouenon[(1)]
Christophe Betoule[(1)], Emmanuelle Delfour[(1)], Shweta Vachhani[(2)] and Balagangadhar Bathula[(2)]

[(1)] Orange Labs, 2 Avenue Pierre Marzin, Lannion, France, ahmed.triki@orange.com
[(2)] AT&T Labs, 200 Laurel Avenue South, Middletown, NJ, USA

Abstract *Given the variety of open-source initiatives, we propose to use GNPy as reference for QoT computation in multi-vendor optical transport networks. Therefore, we develop an API to allow GNPy to dynamically interact with any controller, particularly TransportPCE which uses GNPy to validate computed paths.*

Introduction

Open-source development is a new scope in optical transport network that enables the main players in the field to push their views and needs around challenging topics and issues. Mainly supported by Telecom operators and academia, interoperability is one of the topics addressed by open-source communities in order to cope with the reluctance of equipment suppliers to adopt standardized models where designing their equipment and services and unlock their proprietary network management systems (NMS).

Adopting open and common data models and APIs (Application Programming Interfaces) facilitates the softwarization of the optical layer and paves the way towards a fully-automated network. Moreover, it accelerates the adoption of new functions and innovative solutions by simplifying the testing and the integration processes. In addition to the technical advantages, openness allows operators to adopt multi-sourcing policy and stimulate competition between suppliers when purchasing new equipment and services. In such ecosystem, equipment suppliers are invited to adopt a new business model based on providing stand-alone hardware/software blocks instead of a complete turnkey solution[1].

Many open-source communities are interested in openness and interoperability and they study this subject from different perspectives: open and standard data-model, open-source code for controller implementation and open tools for design and performance estimation. OpenROADM Multi-Source Agreement (MSA)[2] is one of the promising initiative that focuses on proposing vendor-agnostic data models providing disaggregated description of optical equipment, as well as, an optimized and comprehensive view of the topology and the network services. In order to show the strength of these models, TransportPCE (T-PCE)[3] project aims to provide a reference implementation of OpenROADM data-models[4].

T-PCE is an open-source project and a feature of OpenDaylight. Its primary function is to control an optical transport infrastructure using a non-proprietary south bound interface. T-PCE provides also north bound interfaces to interconnect with an Orchestrator, a higher layer Controller and/or other software as showed in Fig. 1.

The interaction between T-PCE and other OpenROADM non-compliant controllers/orchestrators to establish an inter-domain path or support alien wavelength could rise issues concerning the absence of reference to ensure some critical operations such as the estimation of the Quality of Transmission (QoT). In this context, GNPy (Gaussian Noise in Python)[5] could be a favorite candidate to handle this function since it is an open-source tool able to assess optical impairments and provide accurate QoT metric regardless of data model.

This paper highlights the latest developments in T-PCE to use GNPy as reference to check light-path feasibility and perform impairment-aware path computation. In the next section, we describe the proposed API that transforms GNPy from a design tool to a plugin module able to interact with any controller particularly T-PCE. Then, in the last section, we present the T-PCE path computation algorithm and show some results.

T-PCE/GNPy Interface

GNPy is an open-source software that models and designs optical network considering the specifications and settings of equipment as defined by their vendors. By accurately simulating optical propagation, GNPy becomes a potential

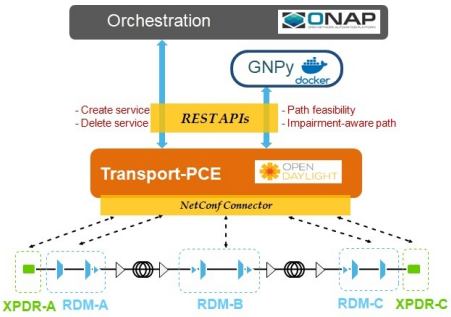

Fig. 1: Interaction between T-PCE and external modules

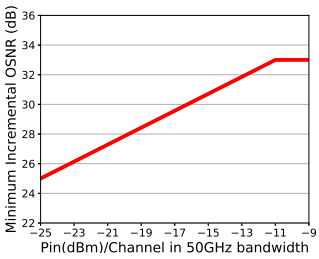

Fig. 2: Noise Mask according to ROADM MSA

alternative to proprietary tools as well as a reference to fairly compare vendors' solutions.

The computation of performance metrics in GNPy is based on Gaussian Model taking into account linear and non-linear impairments. In order to take advantage of the accuracy of GNPy engine and use it in the path computation module of T-PCE, we have developed a REST API allowing real-time interaction between the two open-source software via HTTP requests. The API provides methods to check if a path is optically feasible and to propose an alternative path otherwise. Note that feasibility means that the path's QoT meets the transceiver performance requirements in terms of SNR.

The API of GNPy is mainly based on Flask RESTful framework[6] that provides a lightweight web server gateway. The request/response format of the HTTP request is defined by YANG models available in[7]. The body of the HTTP POST request describes the network topology and the service context. The topology description includes the list of elements (i.e., ROADM, transceivers, amplifiers, fibers, ...) and the connections between them. The service context mainly consists of the source/destination nodes, service requirements and constraints such as bandwidth, transceiver mode and elements to include/exclude in the path. In the case of checking path feasibility, the service context contains also the list of elements composing the path in question. The response of the HTTP POST request returns the list of elements of the path, a boolean field indicating the feasibility of the path and QoT path metrics (i.e., 0.1 nm OSNR, bandwidth OSNR, 0.1 nm SNR and bandwidth SNR). To ensure its portability and availability, the GNPy code and its dependencies are packaged into a Docker container that could run uniformly and consistently across any platform or cloud.

On the T-PCE side, a GNPy interface module is developed to create/analyze the GNPy request/response. The central task of the module consists of matching between the OpenROADM elements (i.e., equipment, topology and service) and those of the GNPy YANG model. The correspondence between the two data models has been possible due to the flexibility of the OpenROADM model which offers three levels of topology abstraction (i.e., "CLLI Network", "Openroadm Network" and "Openroadm Topology"), presenting the different disaggregation level of the topology. Note that the OpenROADM equipment specifications are directly integrated in the GNPy equipment library without going through the API.

The major advantage of the API is that T-PCE (or any other controller) does not need to implement comprehensive QoT estimator but rather relies on GNPy and benefits from the evolution of the tool which avoids duplication of effort. Moreover, in the absence of standard to compute some metrics such as SNR, GNPy could be a credible reference to estimate QoT. This is particularly useful in the case of computing a path crossing many network domains adopting different data models (i.e., multi-domain path).

Path Computation Algorithm and Results

In T-PCE, the Service Handler module (SH) receives service creation/deletion request coming from a higher level controller or an orchestrator via the northbound API. After checking the consistency of the request and its compliance with the OpenROADM service model, the SH triggers the Path Computation Entity (PCE) module by sending a path computation request (Internal API). The request defines hard constraints related to the nodes to be included/excluded in the path, the PCE metric (number of hops is taken by default) and the maximum path latency.

In the first selection round, the PCE computes the k-shortest paths based on Bellman-Ford algorithm, sorts them according to their weights and filters out those which have a number of hops/ latency exceeding the upper limit. Then, as second selection round, it computes the 0.1 nm OSNR of each path until it finds the first one hav-

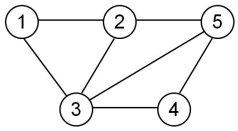

Fig. 3: Five-node network topology

ing an OSNR greater than the receiver OSNR threshold. To compute the OSNR of the path, the PCE uses the Noise Mask specified in the OpenROADM MSA for ILA (In-Line Amplifier) and ROADMs (Fig.2). If the GNPy container is properly connected to T-PCE, the PCE module sends the candidate path to it for validation. To confirm the feasibility of the path, the GNPy engine calculates the 0.1 nm SNR and compares it to the receiver OSNR threshold. The GNPy also rechecks the amplifiers working point and the calculation of the 0.1 nm OSNR.

If the path is not feasible, the PCE sends a new request to GNPy, including only the constraints expressed in the path computation request initiated by the SH. GNPy then tries to calculate a path based on these constraints and provides the result to the PCE. The PCE translates the elements of the proposed path, if it exists, into their corresponding nodes in the non-disaggregated OpenROADM topology (i.e., "Openroadm Network" in the MSA) then, computes its corresponding path in the OpenROADM disaggregated topology (i.e., "Openroadm Topology"). Finally, the PCE forwards the results to the SH. The complete implementation of this algorithm is available on the OpenDayLight Magnesium Release[3].

We consider the topology of Fig. 3, composed of five nodes. Each node is composed of one OpenROADM 100-G transponder attached to a ROADM. The mean ROADM-to-ROADM link length is around 100 km and the fiber loss coefficient is equal to 0.25 dB/km. The amplification of the signal is only performed in the ROADM through the preamplifier and the booster. The output power of the ROADM is fixed to 2 dBm. The network is controlled by a T-PCE instance connected to a GNPy docker. We create several paths having different number of hops (from 1 to 4). Fig.4 shows the performance of lightpaths as function of the number of hops. The dashed dark line shows the OSNR tolerance of the 100G receiver according to OpenROADM MSA[2]. We notice that the 0.1nm OSNR computed by GNPy perfectly corresponds to that computed by T-PCE. The difference between the OSNR and the SNR varies from 1 dB to 1.3 dB. Thus, in case the connection to GNPy docker is not possible, the PCE

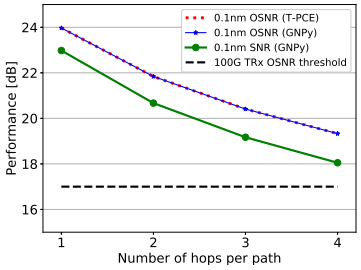

Fig. 4: QoT performance of multi-hop paths

should check the feasibility of the path by applying additional margin (of at least 2 dB) to the computed 0.1nm OSNR in order to take into account the non-linear impairments.

Conclusions

The variety of solutions proposed by open-source communities to address interoperability in optical network rises the need for reference tools to ensure critical operations. In this paper, we propose to use GNPy as a third-party application to handle the QoT estimation since it is model-agnostic and provides accurate assessment of linear and non-linear impairments. Our contribution consists of creating a REST API for GNPy to transform it from a design tool to an interactive module able to connect in real-time with any controller/software. The API is used by T-PCE to improve its impairment-aware path computation algorithm by setting an active exchange between the two modules taking into account a set of constraints. This is the first time that GNPy runs as a third-party plugin module to assist a controller in real-time and it is also the first time that T-PCE delegates one of its features (i.e., computing and checking the feasibility of optical path) to an external application.

References

[1] L. Alahdab *et al.*, "Alien wavelengths over optical transport networks", *IEEE/OSA JOCN*, vol. 10, no. 11, pp. 878–888, 2018.

[2] *Openroadm multi-source agreement (msa) specification*, version 3.0.1, OpenROADM, Jul. 2019.

[3] Transport-PCE. (2020). Gerrit: Opendaylight code deposit website, [Online]. Available: `https://git.opendaylight.org/gerrit/q/project:transportpce`.

[4] A. Triki *et al.*, "Openroadm compliant sdn controller for a full interoperability of the optical transport network", in *ECOC*, IEEE, 2018, pp. 1–3.

[5] J.-L. Auge *et al.*, "Open optical network planning demonstration", in *OFC*, Optical Society of America, 2019, M3Z–9.

[6] Flask-RESTful. (2020). Project documentation, [Online]. Available: `https://flask-restful.readthedocs.io`.

[7] oopt-gnpy. (2020). Github: Telecominfra-project code deposit website, [Online]. Available: `https://github.com/Telecominfraproject/oopt-gnpy`.