

Open Optical Networks

Lab 7

November 29, 2022

uniform switching matrix - lab 06

These exercise sets cover some aspects you will find useful for the final exam software development. This exercises can be part of the material for the final exam questions. You are strongly encouraged to find yourself a solution to the presented problems.

Exercises

Non-uniform Switching Matrix

node_not_full.json contains blocked nodes

Two new json descriptions of the network are given:
"node_full.json" and "node_not_full.json"

1. They contains the information of the switching matrix defined for every node in the network. In the first one all the switching matrices are full, meaning that given a node, the spectral information (light-path) can travel from and to any line connected to the node (except going backwards on the same line). In the second one some of these possibilities are prevented defining not full switching matrices. Modify the **Network** constructor and the method **connect()** in order to properly retrieve the switching matrix information from the description json file.
2. Run the main that evaluates the distribution of the SNR on a list of 100 randomly chosen connections for both the network description json files and compare the two results.
3. In realistic scenarios, the switching matrix varies dynamically due to light-path deployments. In order to reproduce a rough abstraction of this behaviour, suppose that, whenever a wavelength cross a node, from a node "in" to a node "out", it blocks the same "in-out" propagation for the first adjacent wavelengths. E.g., assume that the light-path crosses the node "B" coming from "A" to node "C". If the light-path channel is "3", than after the propagation the channel "2" and "4" cannot cross the node "B" coming from "A" to node "C". Modify the **propagate()** method in the **Node** class in order to include this feature by means of a dynamic

when you have a roudam
you have a limit number of phisical port
that allow you to connect

you have a multiplexer and demultrplexer
in each direction

modification of the switching matrices (be sure that the initial switching matrices are stored in the **Network** object and that they are restored after the streaming).

4. Run again the main that evaluates the distribution of the SNR on a list of 100 randomly chosen connections for both the network description json files and compare the two results with the previous ones.

Transceivers

Given the weighted graph based on the GSNR metric of each path, we want to add the concepts of capacity and transceiver strategy.

1. Implement a method **calculate_bit_rate(path, strategy)** in the **Network** class that evaluates the bit rate R_b supported by a specific path given the corresponding GSNR and the transceiver technology using the following equations:

$$R_b = \begin{cases} 100 \text{ Gbps,} & \text{if } \text{GSNR} \geq 2\text{erfcinv}^2(2\text{BER}_t) \frac{R_s}{B_n} ; \\ 0 \text{ Gbps,} & \text{otherwise} \end{cases} \quad (1)$$

you can know what space initialize

when you are perform the graph of the latency we are using

the prob method

$$R_b = \begin{cases} 0 \text{ Gbps,} & \text{if } \text{GSNR} < 2\text{erfcinv}^2(2\text{BER}_t) \frac{R_s}{B_n} \\ 100 \text{ Gbps,} & \text{if } 2\text{erfcinv}^2(2\text{BER}_t) \frac{R_s}{B_n} \leq \text{GSNR} < \frac{14}{3}\text{erfcinv}^2(\frac{3}{2}\text{BER}_t) \frac{R_s}{B_n} ; \\ 200 \text{ Gbps,} & \text{if } \frac{14}{3}\text{erfcinv}^2(\frac{3}{2}\text{BER}_t) \frac{R_s}{B_n} \leq \text{GSNR} < 10\text{erfcinv}^2(\frac{8}{3}\text{BER}_t) \frac{R_s}{B_n} \\ 400 \text{ Gbps,} & \text{if } \text{GSNR} \geq 10\text{erfcinv}^2(\frac{8}{3}\text{BER}_t) \frac{R_s}{B_n} \end{cases} \quad (2)$$

$$R_b = 2R_s \log_2 \left(1 + \text{GSNR} \cdot \frac{R_s}{B_n} \right) \text{ Gbps} ; \quad (3)$$

where (1) is for the **fixed-rate** transceiver strategy assuming PM-QPSK modulation, (2) is for the **flex-rate** transceiver strategy assuming the availability of PM-QPSK (100Gbps), PM-8-QAM (200Gbps) and PM-16QAM (400Gbps) modulations, given a BER_t of 10^{-3} . In conclusion, (3) is the maximum theoretical Shannon rate with an ideal Gaussian modulation. R_s is the symbol-rate of the light-path that can be fixed to 32 GHz and B_n is the noise bandwidth (12.5 GHz).

2. Add the attribute **transceiver** to the class **Node**. This attribute can be **fixed-rate**, **flex-rate**, **shannon** and must be read from the network description json file. If the transceiver is not provided in the description file, it has to be set as **fixed-rate**.
3. Modify the **stream()** method of the **Network** class that has to call the **calculate_bit_rate(path, strategy)** once the path for the connection

is given and using the transceiver attribute value of the first node in the path. If the path does not reach the minimum GSNR requirement for the specified transceiver strategy (zero bit rate case), the connection has to be rejected. Add the attribute **bit_rate** to the **Connection** class that stores the assigned bit rate R_b .

4. Run the main that evaluates the distribution of the SNR on a list of 100 randomly chosen connections for the three newly provided network description json files and plot the histogram of the accepted connections bit rates calculating the overall average. Also calculate the total capacity allocated into the network. Compare the three results obtained for the three different transceiver strategies.