Open Optical Networks
Lab 4

May 6, 2021

The aim of these exercises is to use the software abstraction of the network described in the last laboratory exercise set (Lab 3) and simulate few cases of signal propagation.

Exercises

1. Set the dataframe constructed in exercise 5 of Lab 3 as an attribute of the network called 'weighted_paths'.

2. Define a method **find_best_snr()** in the class **Network** that, given a pair of input and output nodes, returns the path that connects the two nodes with the best (highest) signal to noise ratio introduced by the signal propagation.

3. Define a method **find_best_latency()** in the class **Network** that, given a pair of input and output nodes, returns the path that connects the two nodes with the best (lowest) latency introduced by the signal propagation.

4. Define the class **Connection** that has the attributes:

   - **input**: string
   - **output**: string
   - **signal_power**: float
   - **latency**: float
   - **snr**: float

   The attributes **latency** and **snr** have to be initialized to zero.

5. Define the method **stream** in the class **Network** that, for each element of a given list of instances of the class **Connection**, sets its **latency** and **snr** attribute. These values have to be calculated propagating a **SignalInformation** instance that has the path that connects the **input** and the **output** nodes of the connection and that is the best snr or latency

path between the considered nodes. The choice of latency or snr has to be made with a label passed as input to the stream function. The label default value has to be set as latency.

6. Create a main that constructs the network defined by 'nodes.json' and runs its method **stream** over 100 connections with **signal_power** equal to one and the input and output nodes randomly chosen. This run has to be performed in turn for latency and snr path choice. Accordingly, plot the distribution of all the latencies or the snrs.

7. Modify the class **Line** such that it includes an attribute **state** that can assume the values 'free' or 'occupied' (initialize it as 'free'). This attribute shows if a connection is already occupying that line. Modify accordingly the **find_best_latency()** and **find_best_snr()** methods that have to return the best available path, meaning that all the lines within the path have to be 'free'. Moreover, modify the **stream** network method such that, if there are not any available path between the input and the output nodes of a connection, the resulting snr and latency have to be set to zero and 'None', respectively. Run again the main of the previous exercise with the snr path choice.