NetXPTO - NetPlanner

9 de Abril de 2019

Conteúdo

1 Heuristic Models **3**

1.1 CAPEX . . . 4

1.1.1 Opaque without Survivability . . . 8

1.1.2 Opaque with 1+1 Protection . . . 50

1.1.3 Transparent without Survivability . . . 75

2 Transparent transport mode **100**

2.1 Concepts . . . 101

2.1.1 Network Nodes . . . 101

2.1.2 Network Links . . . 101

2.1.3 Network Topology . . . 102

2.1.4 Lightpaths . . . 102

2.2 System inputs . . . 102

2.3 System signals . . . 103

2.4 Type signals structures . . . 103

2.4.1 LogicalTopology . . . 103

2.4.2 PhysicalTopology . . . 104

2.4.3 DemandRequest . . . 106

2.4.4 PathRequest . . . 106

2.4.5 PathRequestRouted . . . 106

2.4.6 DemandRequestRouted . . . 107

2.5 Blocks input parameters and signals . . . 108

2.6 Blocks state variables and output signals . . . 108

2.7 Network example . . . 109

2.7.1 Physical topology matrix . . . 109

2.7.2 Logical topology matrix . . . 110

2.7.3 Traffic scenario . . . 110

2.7.4 Theorectical resolution . . . 111

2.7.5 Demand 1 . . . 116

2.7.6 Demand 2 . . . 120

2.7.7 Demand 3 125
2.7.8 Demand 4 129
2.7.9 Demand 5 134
2.7.10 Demand 6 138
2.7.11 Final Report 141
2.7.12 Transparent with 1+1 Protection 144
2.7.13 Translucent without Survivability 171
2.7.14 Translucent with 1+1 Protection 196

Capítulo 1

Heuristic Models

Heuristic algorithms are used in this report with an objective to minimize the total CAPEX of the network. It is important to have these comparison results with the linear programming ones, because these formulations are faster than the optimal methods and can generate also a near optimal solution. Another advantage of the heuristic approach is that heuristic solutions leads to good performances in practical network scenarios when we present a sufficiently feasible solution, instead of an optimal solution.
In order to get the total network cost for the reference network, the CAPEX is calculated by routing and grooming heuristic algorithms implemented in Java and it is considered that all network equipment is bidirectional. These algorithms are tested in a network design software called Net2Plan.
This chapter consists in demonstrating how the matrices are created, how the heuristic algorithms work and analyzing the results. It is divided in six subsections and the results differ into three different transport modes: opaque (link-by-link grooming method), transparent (single-hop grooming method) and translucent (multi-hop grooming method). Each one of these transport methods are also distinguished and compared by the possibility of being without survivability or with 1+1 protection and for the cases of low, medium and high traffic in the network.

1.1 CAPEX

Student Name	:	Pedro Coelho (01/03/2018 -)
Goal	:	Implement of the heuristic model to obtain the best possible CAPEX of a given network.

The total CAPEX of a network, as it was already described in **??**, is the sum between two differentiated costs. Firstly, the link cost depends on the link length, which has integrated components such as OLTs, transceivers and amplifiers and the node cost depends on the traffic intensity by each node.

In order to get the results for the heuristic approach, some algorithms are used which try to obtain the most near optimal solution for the six cases detailed in this chapter. Then, it will be applied a cost report with all the detailed information about the costs of the network. The final CAPEX depends on the transport mode (opaque, transparent and translucent), possibility of the network having a dedicated 1+1 protection scheme or not and the network traffic (low - 0.5 Tbit/s, medium - 5 Tbit/s and high - 10 Tbit/s).

To calculate the total network cost it has to be considered the links cost and the nodes cost. The CAPEX value of a network, C_C, in monetary units (e.g. euros, or dollars), is calculated by the equation 1.1

$$C_C = C_L + C_N \tag{1.1}$$

where

- $C_L \rightarrow$ Link cost in monetary units (e.g. euros, or dollars)
- $C_N \rightarrow$ Node cost in monetary units (e.g. euros, or dollars)

On the first hand, the links' cost, C_L, in monetary units (e.g. euros, or dollars), is calculated by the equation 1.2. If the length of the link is longer, the resulting costs will be higher due to of the necessity of having more components which carry all the traffic from all origin nodes to all destination nodes

$$C_L = \sum_{i=1}^{N} \sum_{j=i+1}^{N} L_{ij} \left(2\gamma_0^{OLT} + 2\gamma_1^{OLT} \tau W_{ij} + N_{ij}^R c^R \right) \tag{1.2}$$

where

- $i \rightarrow$ Index for start node of a physical link
- $j \rightarrow$ Index for end node of a physical link
- $N \rightarrow$ Total number of nodes, N $\in \mathbb{N}$
- $L_{ij} \rightarrow$ Binary variable indicating if link between the nodes i and j is used, $L_{ij} \in 0, 1$

- $\gamma_0^{OLT} \rightarrow$ OLT cost in monetary units (e.g. euros, or dollars)
- $\gamma_1^{OLT} \rightarrow$ Transponder cost in monetary units (e.g. euros, or dollars)
- $\tau \rightarrow$ Line bit-rate
- $W_{ij} \rightarrow$ Number of optical channels in link $i\ j$
- $N_{ij}^{R} \rightarrow$ Number of optical amplifiers in link $i\ j$
- $c^{R} \rightarrow$ Optical amplifiers cost in monetary units (e.g. euros, or dollars)

The line bit-rate that is used in this report has the value of 100. It represents the separation between regeneration stages in km. On the other hand, the nodes' cost, C_N, in monetary units (e.g. euros, or dollars), is calculated by the equation 1.3

$$C_N = C_{EXC} + C_{OXC} \tag{1.3}$$

where

- $C_{EXC} \rightarrow$ Electrical node cost in monetary units (e.g. euros, or dollars)
- $C_{OXC} \rightarrow$ Optical node cost in monetary units (e.g. euros, or dollars)

As all the nodes have an electrical part and an optical part, it is necessary to calculate both with their respective formulas and sum the results. The constitution of the electrical node part can be seen in the Integer Linear Programming section **??**. The electrical nodes' cost, C_{EXC}, in monetary units (e.g. euros, or dollars), is given by the equation 1.4

$$C_{EXC} = \sum_{n=1}^{N} N_{exc,n} \left(\gamma_{e0} + \sum_{c=-1}^{B} \gamma_{e1,c} P_{exc,c,n} \right) \tag{1.4}$$

where

- $N \rightarrow$ Total number of nodes, N $\in \mathbb{N}$
- $N_{exc,n} \rightarrow$ Binary variable indicating if node n is used, $N_{exc,n} \in 0, 1$
- $\gamma_{e0} \rightarrow$ EXC cost in monetary units (e.g. euros, or dollars)
- $\gamma_{e1,c} \rightarrow$ EXC port cost in monetary units (e.g. euros, or dollars) with bit-rate B and with a given transceiver reach
- $P_{exc,c,n} \rightarrow$ Number of ports of the electrical switch
- $B \rightarrow$ A natural number corresponding to the maximum index of short-reach ports, see table below

Index	Bit rate
-1	100 Gbits/s line bit-rate (long-reach port)
0	1.25 Gbits/s tributary bit-rate (short-reach port)
1	2.5 Gbits/s tributary bit-rate (short-reach port)
2	10 Gbits/s tributary bit-rate (short-reach port)
3	40 Gbits/s tributary bit-rate (short-reach port)
4	100 Gbits/s tributary bit-rate (short-reach port)

Tabela 1.1: Table with index and your corresponding bit rate

The constitution of the optical node part can be seen in the Integer Linear Programming section **??**. Finally, the optical nodes' cost, C_{OXC}, in monetary units (e.g. euros, or dollars), is given by the equation 1.5

$$C_{OXC} = \sum_{n=1}^{N} N_{oxc,n} \left(\gamma_{o0} + \gamma_{o1} P_{oxc,n}\right) \tag{1.5}$$

where

- $N \rightarrow$ Total number of nodes, N $\in \mathbb{N}$
- $N_{oxc,n} \rightarrow$ Binary variable indicating if node n is used, $N_{oxc,n} \in 0, 1$
- $\gamma_{o0} \rightarrow$ OXC cost in monetary units (e.g. euros, or dollars)
- $\gamma_{o1} \rightarrow$ OXC port cost in monetary units (e.g. euros, or dollars)
- $P_{oxc,n} \rightarrow$ Number of ports of the optical switch

After all the formulas needed to calculate the CAPEX of a network are demonstrated above, it is also essential to have pre-defined the costs of the network equipments. In the table 1.2 it is shown the cost, in euros, of all the equipments with their symbols used in the respective formulas.

Equipment	Symbol	Cost
OLT without transponders	γ_0^{OLT}	15000 €
Transponder	γ_1^{OLT}	5000 €/Gb
Unidirectional Optical Amplifier	c^R	4000 €
EXC	γ_{e0}	10000 €
OXC	γ_{o0}	20000 €
EXC Port for line ports	$\gamma_{e1,-1}$	1000 €/Gb/s
EXC Port for ODU0	$\gamma_{e1,0}$	8 €/Gb/s
EXC Port for ODU1	$\gamma_{e1,1}$	6 €/Gb/s
EXC Port for ODU2	$\gamma_{e1,2}$	3 €/Gb/s
EXC Port for ODU3	$\gamma_{e1,3}$	1.5 €/Gb/s
EXC Port for ODU4	$\gamma_{e1,4}$	1 €/Gb/s
OXC Port	γ_{o1}	2500 €/porto

Tabela 1.2: Table with costs

1.1.1 Opaque without Survivability

Student Name	:	Élio Coelho (01/10/2018 -)
	:	Pedro Coelho (01/03/2018 - 30/06/2018)
Goal	:	Implement the Heuristic model for the opaque transport mode without survivability.

In the opaque transport mode (link-by-link approach), the lightpath entering any intermediate node is necessarily terminated, i.e., there are performed OEO (optical-electrical-optical) conversions at every intermediate node since the origin to the destination node. These conversions are used for every wavelength at every node.

Contrary to the opaque with dedicated 1+1 protection technique, the opaque without survivability technique does not have a backup path, so if there is a network failure it is more likely to suffer large data losses, which consequently leads to higher network costs. However, the CAPEX will be significantly lower, because that not includes a secondary path that will increase several network elements.

After the creation of the matrices and the network topology, it is necessary to apply the routing and grooming algorithms created. In the end, a cost report algorithm will be applied to visualize the network CAPEX results for the network in question.

Firstly, in the opaque transport mode, the optical node cost is 0 because all the ports in the network are electrical. Consequently, to calculate the nodes' cost in this transport mode it only has to be considered the electrical nodes' cost:

- $N_{OXC,n}$ = 0, $\forall$ n
- $N_{EXC,n}$ = 1, $\forall$ n that process traffic

As previously mentioned, equation 1.6 refers to the number of long-reach ports of the electrical switch with bit-rate -1 in node n, $P_{exc,-1,n}$, i.e., the number of line ports of node n which can be calculated as

$$P_{exc,-1,n} = \sum_{j=1}^{N} w_{nj} \tag{1.6}$$

where w_{nj} is the number of optical channels between node n and node j.

As previously mentioned, equation 1.7 refers to the number of short-reach ports of the electrical switch with bit-rate c in node n, $P_{exc,c,n}$, i.e., the number of tributary ports with bit-rate c in node n which can be calculated as

$$P_{exc,c,n} = \sum_{d=1}^{N} D_{nd,c} \tag{1.7}$$

where $D_{nd,c}$ are the client demands between nodes n and d with bit rate c.

In this case there is the following particularity:

- When n=j, the value of client demands is always zero, i.e, $D_{nn,c} = 0$.

To implement this heuristic approach there are used algorithms made in Java in a programming software called Eclipse and they are tested in an open-source network program called Net2Plan. In the Net2Plan guide section **??** there is an explanation on how to use and test them in this network planner.

In the next pages it will be described all the steps performed to obtain the final results in the opaque transport mode without survivability. In the figure below 1.1 it is shown a fluxogram with the description of this transport mode approach.

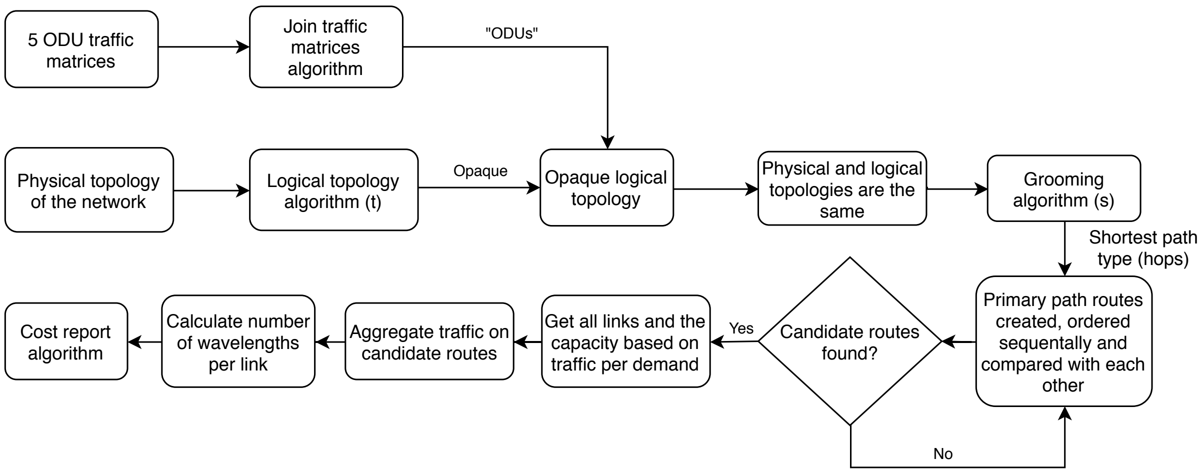

Figura 1.1: Fluxogram with the steps performed in the opaque without survivability transport mode approach.

Creation and join the traffic matrices

The first step is to create the traffic matrices based on the reference network **??**. In order to create the 5 traffic matrices in Net2Plan it is necessary the length of all the links and the total traffic used in this network, so later it is needed to define in Net2Plan the length in all end nodes and the total traffic depends on the value of traffic used (low traffic - 0.5 Tbit/s, medium traffic - 5 Tbit/s and high traffic - 10 Tbit/s). As you can see in the figure below, it is defined the path of the 5 ODUs and they will be aggregated in just one single ODU, making it possible to join all the demands in just one file and load it later into the network. This final resulting ODU joins the multiple traffic demands from all the traffic matrices previously created and, of course, the traffic demands will depend on the values used on the creation of the matrices (low, medium and high traffic).

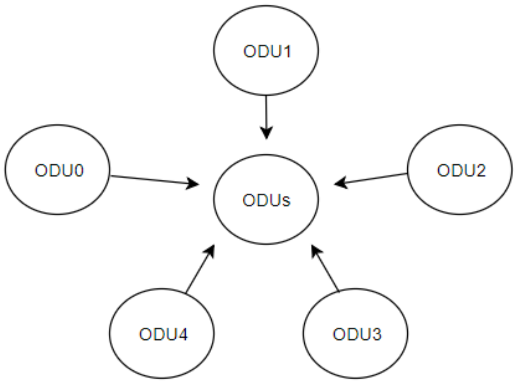

Figura 1.2: Join the 5 ODU traffic matrices into 1 single file "ODUs". The 5 traffic demands from the traffic matrices previously created are joined into 1 file to load it later on Net2Plan.

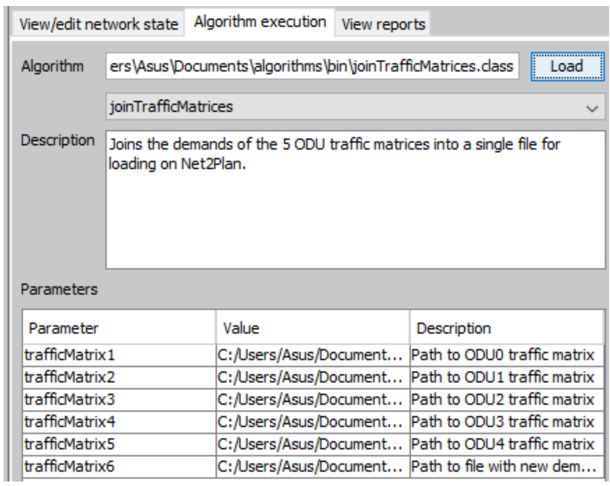

Figura 1.3: Load of the join traffic matrices algorithm for the opaque transport mode on Net2Plan. It is defined the 5 paths to load the 5 ODU traffic matrices and the last path is the one where will be saved the file that joins all 5 the traffic demands.

Creation of the physical topology

The next step is to create the allowed physical topology of the network in Net2Plan. This network consists in 6 nodes and 8 bidirectional links. It is now also possible to define the length in all links. In the figure below it is shown the allowed physical topology in this transport mode.

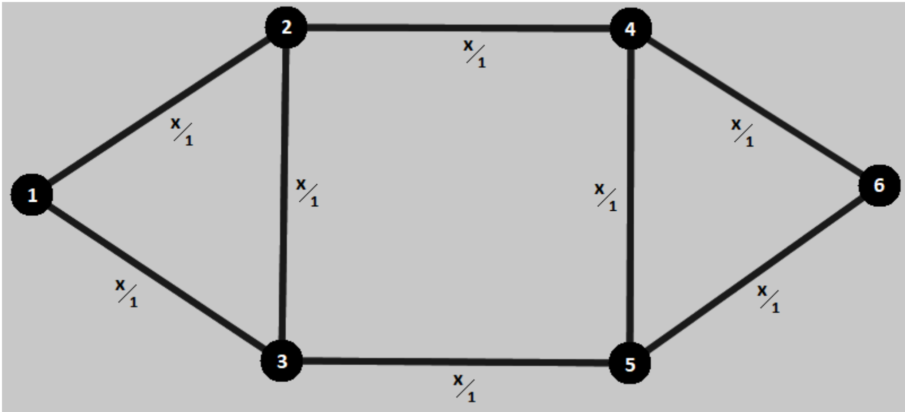

Figura 1.4: Allowed physical topology. The allowed physical topology is defined by the duct and sites in the field. It is assumed that each duct supports up to 1 bidirectional transmission system and each site supports up to 1 node.

Creation of the logical topology

It is now time to create the allowed logical topology. A network topology represents how the links and the nodes of the network interconnect with each other and the logical topology algorithm creates the logical topology on another layer. In the opaque transport mode the physical and logical topologies are the same, so it is needed to add a new layer from the lower layer (default layer). The lower layer is the physical layer of the network and it is now created a new upper layer which is the logical layer of the network and represents the logical topology of the opaque transport mode. The allowed physical and optical topologies, the logical topologies for all ODUs and the resulting physical topology is shown in the next section below 1.1.1 for the three traffic scenarios. It is shown below three figures with the code in Java of the creation of the network logical topology, the load of the logical topology algorithm in Net2Plan and the resulting allowed optical topology for the opaque transport mode without survivability.

```java
if (netPlan.isSingleLayer() && logicalTopology.equalsIgnoreCase("Opaque"))
{

    lowerLayer = netPlan.getNetworkLayerDefault();
    upperLayer = netPlan.addLayerFrom(lowerLayer);
    netPlan.setRoutingType(RoutingType.HOP_BY_HOP_ROUTING, upperLayer);
    lowerLayer.setName("Physical Topology");
    upperLayer.setName("Logical Topology Opaque");
    upperLayer.setDescription("Opaque Logical Topology");

}
```

Figura 1.5: Java code of the logical topology approach for the opaque transport mode. The logical layer is created from the physical layer, as in this transport mode they are the same. The new layer is now the opaque logical topology of the network.

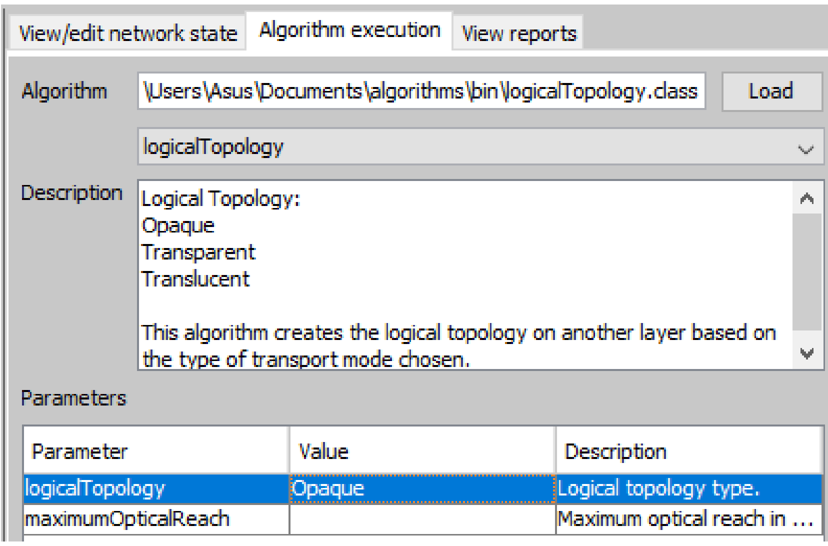

Figura 1.6: Load of the logical topology algorithm for the opaque transport mode.

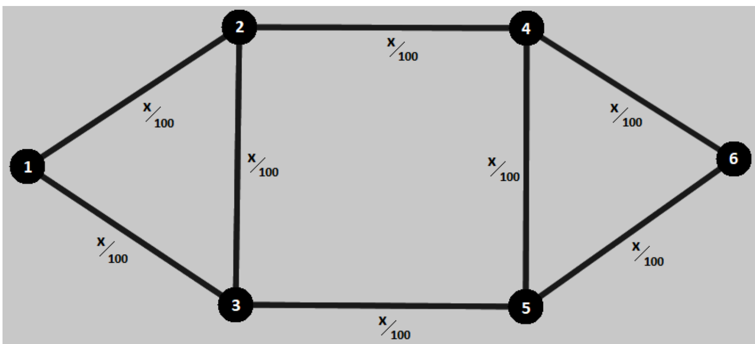

Figura 1.7: Allowed optical topology. It is assumed that each transmission system supports up to 100 optical channels.

Creation of routes and aggregation of traffic

After a network topology is created, it is now time to set the routing algorithm. In the opaque without survivability transport mode the routing algorithm starts with going through all the demands, create bidirectional routes (in this case the primary paths) based on the shortest path Dijkstra algorithm and then search the candidate routes for the respective demand. In this report it is used the shortest path type in hops. These routes are ordered sequentially and the shortest one per each demand is the primary path. The demands from the lower layer are removed and then saved in the upper layer. After this step, it is needed to set the traffic demands into the candidate routes that will integrate the network.

```
case "Logical Topology Opaque":

        for (Demand d : netPlan.getDemands(lowerLayer)) {

            boolean odd=true;
            int counter=0;

            Set<Route> droutes = d.getRoutes();
            System.out.println("droutes: " + droutes.size());

            for(Route c: droutes) {
                counter++;
                boolean jump=false;

                if(odd) {
                    c.setCarriedTraffic(d.getOfferedTraffic(), d.getOfferedTraffic());
                    save=c;
                    odd=false;
                    System.out.println("Roots");
                }
```

Figura 1.8: Creation of routes and aggregation of traffic for the opaque without survivability transport mode. The candidate routes are searched by the shortest path type method and the offered traffic demands are set into these routes.

Function	Definition
netPlan.getDemands(lowerLayer)	Returns the array of demands for the lower layer.
d.getRoutes()	Returns all the routes associated to the demand "d".
c.setCarriedTraffic()	Sets the route carried traffic and the occupied capacity in the links, setting it up to be the same in all links.
d.getOfferedTraffic()	Returns the offered traffic of the demand "d".

Tabela 1.3: Table with the description of the main functions in the creation of routes and aggregation of traffic in the grooming algorithm.

Calculation of the number of wavelengths per link

The final step of the routing and grooming algorithms is to calculate the number of wavelengths per link for the whole network. This is the last and an important step because with the number of wavelengths per link in the network, it is possible to calculate other network components. In the opaque transport mode, as in the figure below shows, the algorithm starts with going through all the links and getting the capacity based on the traffic per demand. The total carried traffic in the link including protection and non-protection segments will be divided by the wavelength capacity and it is now possible to obtain the number of wavelengths per link.

```java
Link p;

for(long e:linkIds) {
    p=netPlan.getLinkFromId(e);
    double sumTraffic = p.getCarriedTrafficNotIncludingProtectionSegments() + p.getReservedCapacityForProtection();
    int nw = (int) (Math.ceil(sumTraffic/wavelengthCapacity));
    String numberWavelengths = String.valueOf(nw);
    p.setCapacity(nw*wavelengthCapacity);
    p.setAttribute("nW", numberWavelengths);
}
break;
```

Figura 1.9: Calculation of the number of wavelengths per link for the opaque transport mode. The link capacity is based on the traffic per demand.

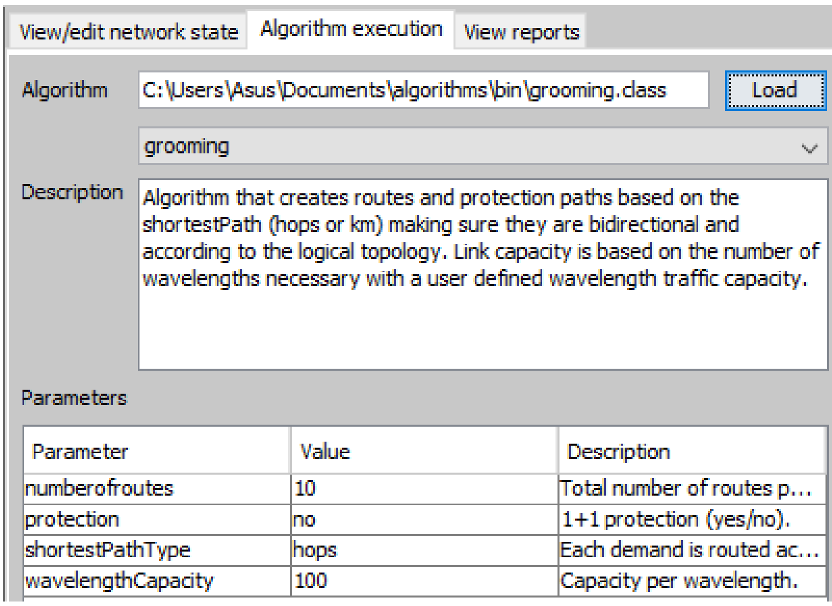

Figura 1.10: Load of the grooming algorithm for the opaque without survivability transport mode. The total number of routes per demand is set to 10, the user can define if the model is with or without protection, the shortest path type is set to "hops" and the capacity per wavelength is used 100 optical channels.

Network cost report

In order to obtain the network CAPEX results, the formulas needed to calculate the network elements and that are demonstrated previously in the beginning of this section 1.1.1 were "translated" into Java code in a cost report algorithm. This algorithm can be loaded in Net2Plan and calculates and shows in tables the network CAPEX and also the per-link and per-node information with more details. In the opaque transport mode the optical node cost is 0 because all the network ports are electrical, so it only has to be considered the electrical nodes' costs and the electrical links' costs.

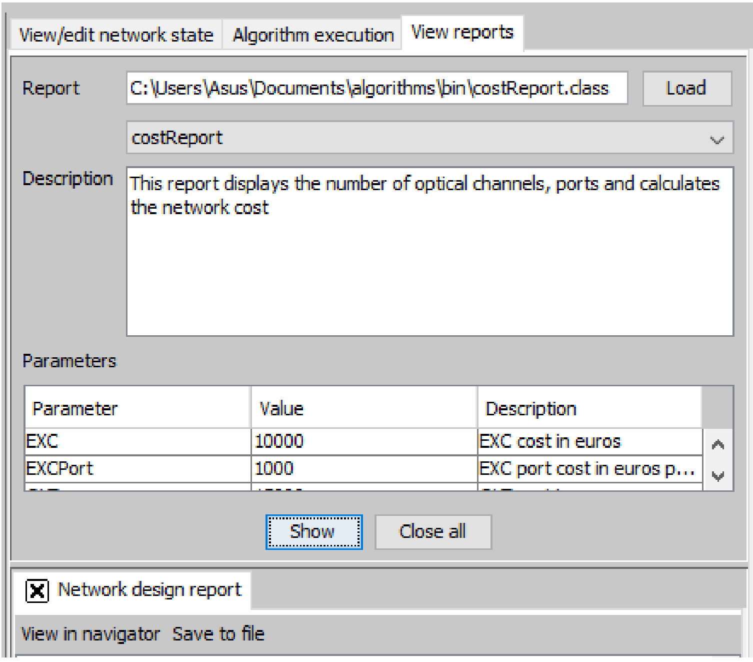

Figura 1.11: Load of the cost report algorithm on Net2Plan. The result view is an HTML page with the network optical and electrical components and their costs.

Result description

It is already known all the necessary formulas to obtain the CAPEX value for the reference network **??**. As described in the subsection of the network traffic **??**, it is necessary to obtain three different values of CAPEX for the low (0.5 Tbit/s), medium (5 Tbit/s) and high (10 Tbit/s) traffic. It is used a network software program called Net2Plan which can design the traffic matrices, create all the network topologies, simulate the algorithms into the network implemented in the programming software called Eclipse and analyze the results obtained. In this chapter will be demonstrated the results by Vasco's heuristics from 2016. In each of the three traffic scenarios, it will be shown the network topologies followed by the table with the CAPEX value of the network.

Low Traffic Scenario:

In this scenario we have to take into account the traffic calculated in **??**. In a first phase we will show the various existing topologies of the network. The first are the allowed topologies, physical and optical topologies, the second are the logical topology for all ODUs and finally the resulting physical topology.

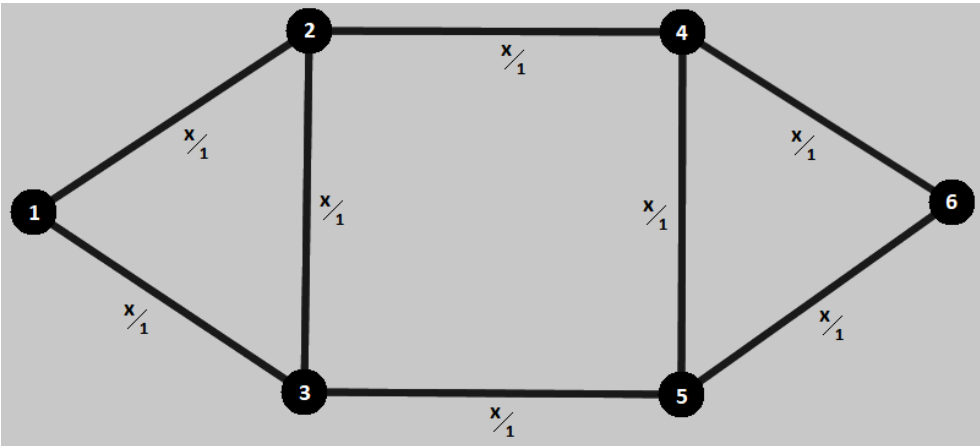

Figura 1.12: Allowed physical topology. The allowed physical topology is defined by the duct and sites in the field. It is assumed that each duct supports up to 1 bidirectional transmission system and each site supports up to 1 node.

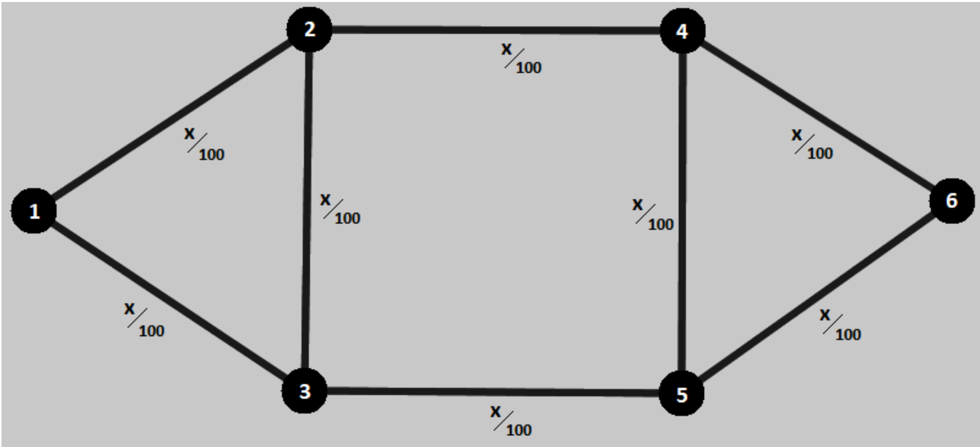

Figura 1.13: Allowed optical topology. The allowed optical topology is defined by the transport mode (opaque transport mode in this case). It is assumed that each transmission system supports up to 100 optical channels.

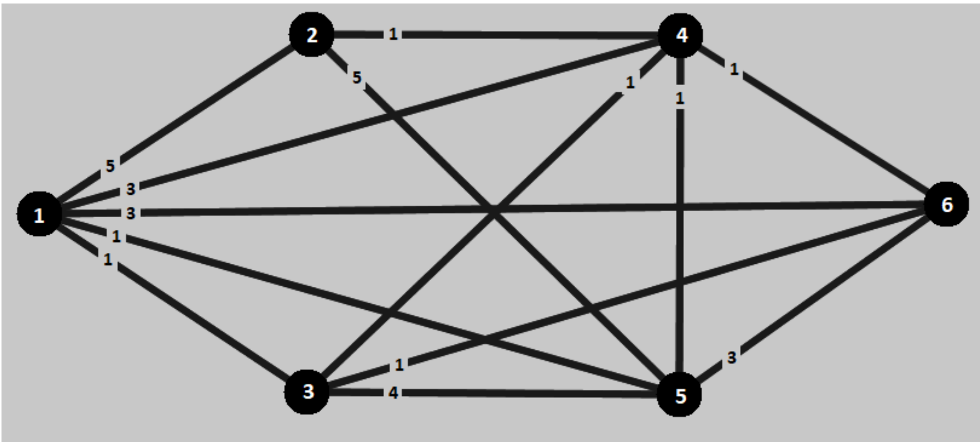

Figura 1.14: ODU0 logical topology defined by the ODU0 traffic matrix.

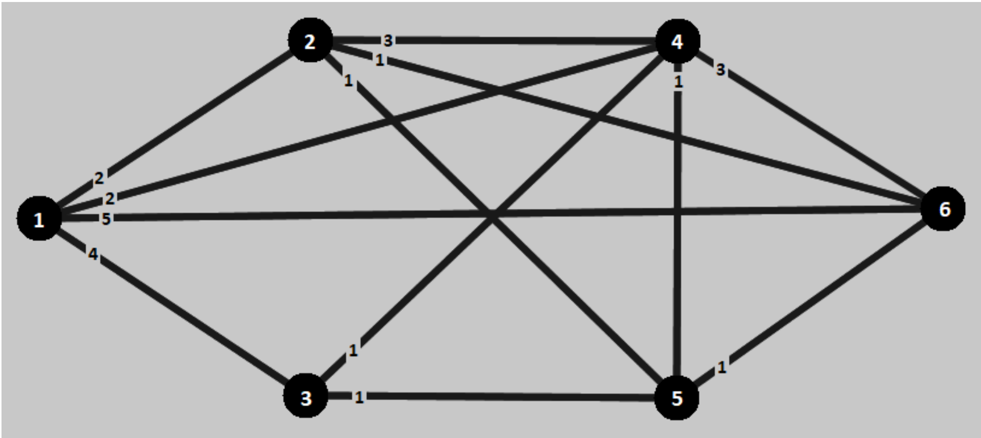

Figura 1.15: ODU1 logical topology defined by the ODU1 traffic matrix.

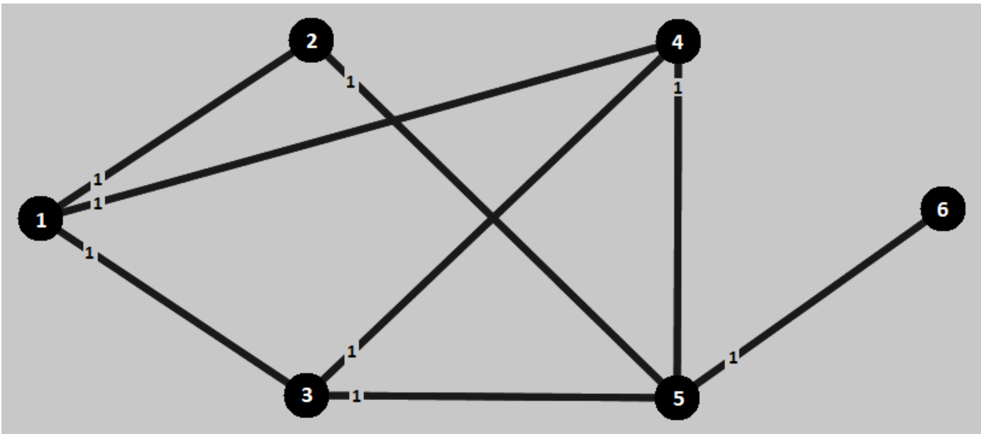

Figura 1.16: ODU2 logical topology defined by the ODU2 traffic matrix.

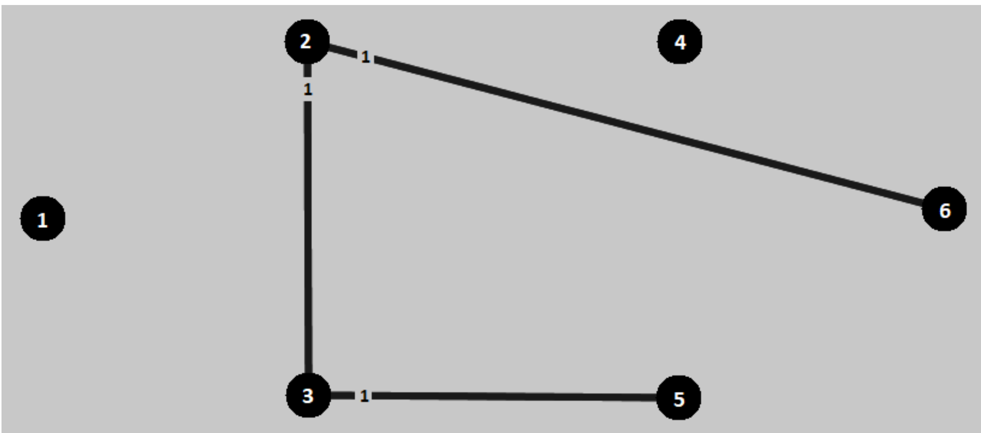

Figura 1.17: ODU3 logical topology defined by the ODU3 traffic matrix.

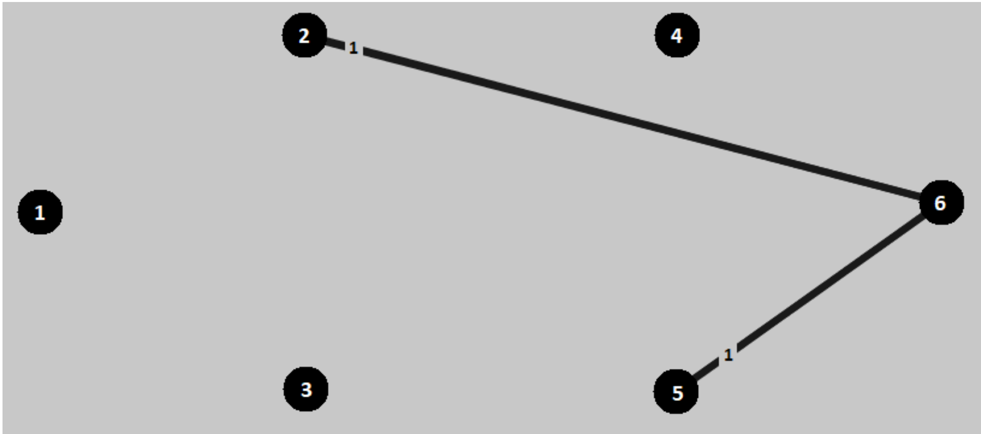

Figura 1.18: ODU4 logical topology defined by the ODU4 traffic matrix.

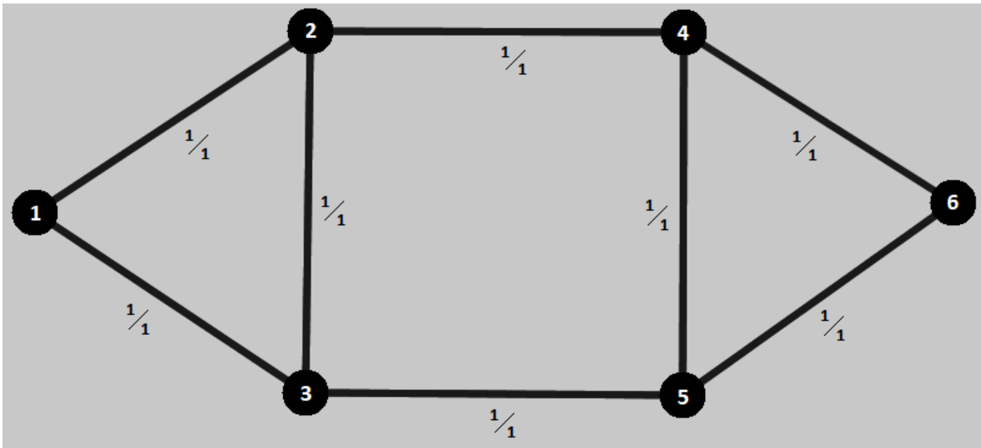

Figura 1.19: Physical topology after dimensioning.

Following all the steps mentioned in the **??**, applying the routing and grooming heuristic algorithms in the Net2Plan software and using all the data referring to this scenario, the obtained result for the Vasco's heuristics can be consulted in the following table 1.4.

All the values calculated in the previous table were obtained through the equations 1.2 and 1.3 referred to in section 1.1, but for a more detailed analysis we created table 1.5 where we can see how all the parameters are calculated individually.

CAPEX of the Network						
			Quantity	Unit Price	Cost	Total
Link Cost	OLTs		16	15 000 €	240 000 €	12 020 000 €
	100 Gbits/s Transceivers		23	5 000 €/Gbit/s	11 500 000 €	
	Amplifiers		70	4 000 €	280 000 €	
Node Cost	Electrical	EXCs	6	10 000 €	60 000 €	2 362 590 €
		ODU0 Ports	60	10 €/port	600 €	
		ODU1 Ports	50	15 €/port	750 €	
		ODU2 Ports	16	30 €/port	480 €	
		ODU3 Ports	6	60 €/port	360 €	
		ODU4 Ports	4	100 €/port	400 €	
		Line Ports	23	100 000 €/port	2 300 000 €	
	Optical	OXCs	0	20 000 €	0 €	
		Ports	0	2 500 €/port	0 €	
Total Network Cost						14 382 590 €

Tabela 1.4: Table with detailed description of CAPEX of Vasco's 2016 results.

	Equation used to calculate the cost
OLTs	$2\sum_{i=1}^{N}\sum_{j=i+1}^{N} L_{ij}\gamma_0^{OLT}$
Transceivers	$2\sum_{i=1}^{N}\sum_{j=i+1}^{N} L_{ij}w_{ij}\gamma_1^{OLT}\tau$
Amplifiers	$2\sum_{i=1}^{N}\sum_{j=i+1}^{N} L_{ij}N_{ij}^{R}c^{R}$
EXCs	$\sum_{n=1}^{N} N_{exc,n}\gamma_{e0}$
ODU0	$\sum_{n=1}^{N}\sum_{d=1}^{N} N_{exc,n}D_{nd,0}\gamma_{e1,0}$
ODU1	$\sum_{n=1}^{N}\sum_{d=1}^{N} N_{exc,n}D_{nd,1}\gamma_{e1,1}$
ODU2	$\sum_{n=1}^{N}\sum_{d=1}^{N} N_{exc,n}D_{nd,2}\gamma_{e1,2}$
ODU3	$\sum_{n=1}^{N}\sum_{d=1}^{N} N_{exc,n}D_{nd,3}\gamma_{e1,3}$
ODU4	$\sum_{n=1}^{N}\sum_{d=1}^{N} N_{exc,n}D_{nd,4}\gamma_{e1,4}$
Line	$\sum_{n=1}^{N}\sum_{j=1}^{N} N_{exc,n}w_{nj}\gamma_{e1,-1}$
OXCs	For opaque transport mode this parameter is always zero.
P_{oxc}	For opaque transport mode this parameter is always zero.
CAPEX	The final cost is calculated by summing all previous results.

Tabela 1.5: Table with description of calculation

Medium Traffic Scenario:

In this scenario we have to take into account the traffic calculated in **??**. In a first phase we will show the various existing topologies of the network. The first are the allowed topologies, physical and optical topologies, the second are the logical topology for all ODUs and finally the resulting physical topology.

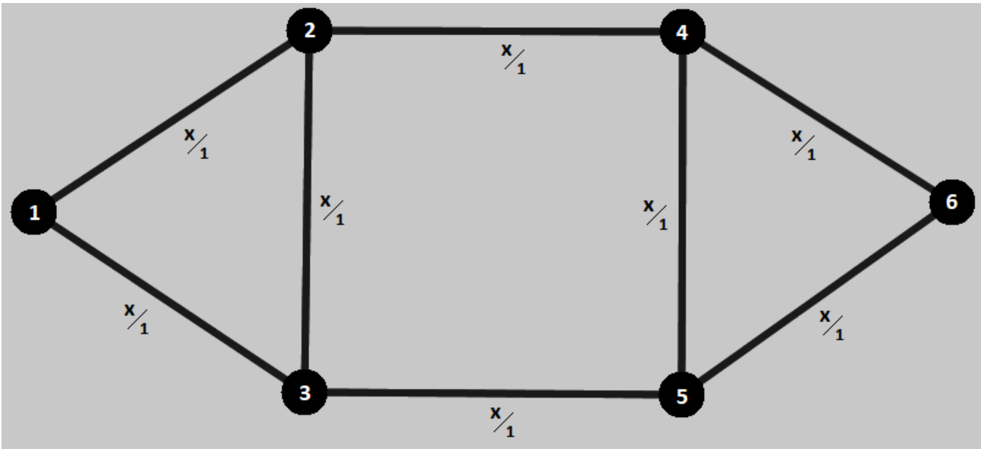

Figura 1.20: Allowed physical topology. The allowed physical topology is defined by the duct and sites in the field. It is assumed that each duct supports up to 1 bidirectional transmission system and each site supports up to 1 node.

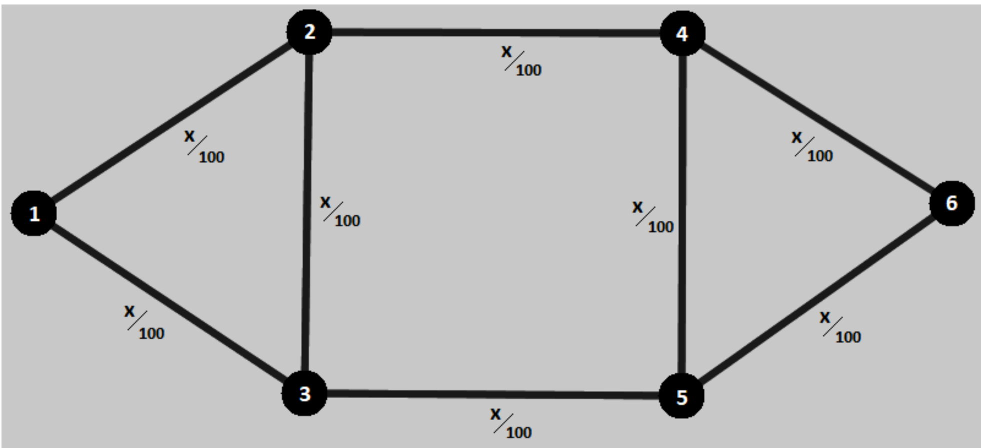

Figura 1.21: Allowed optical topology. The allowed optical topology is defined by the transport mode (opaque transport mode in this case). It is assumed that each transmission system supports up to 100 optical channels.

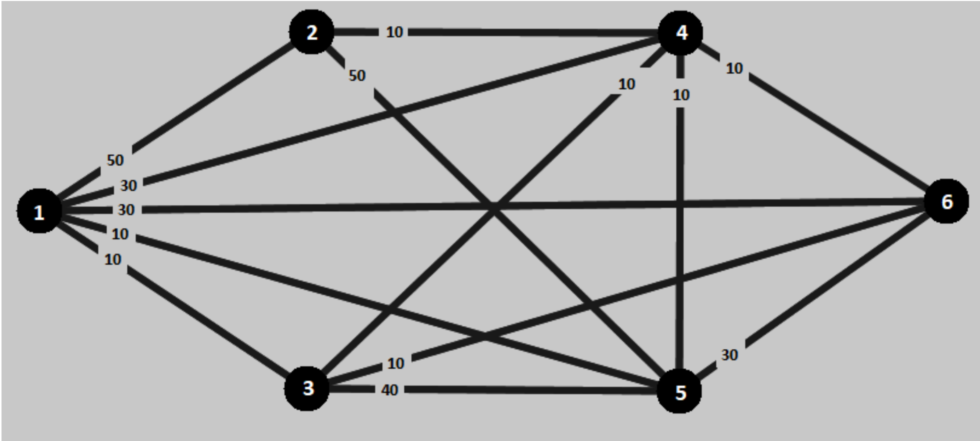

Figura 1.22: ODU0 logical topology defined by the ODU0 traffic matrix.

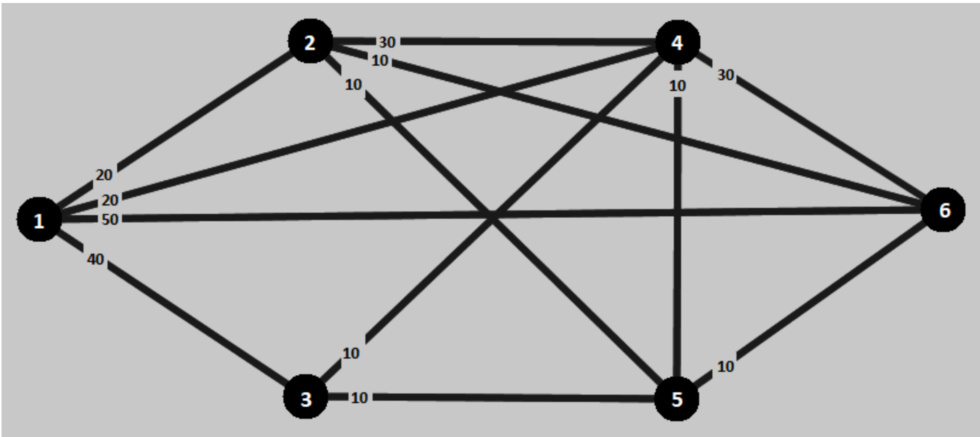

Figura 1.23: ODU1 logical topology defined by the ODU1 traffic matrix.

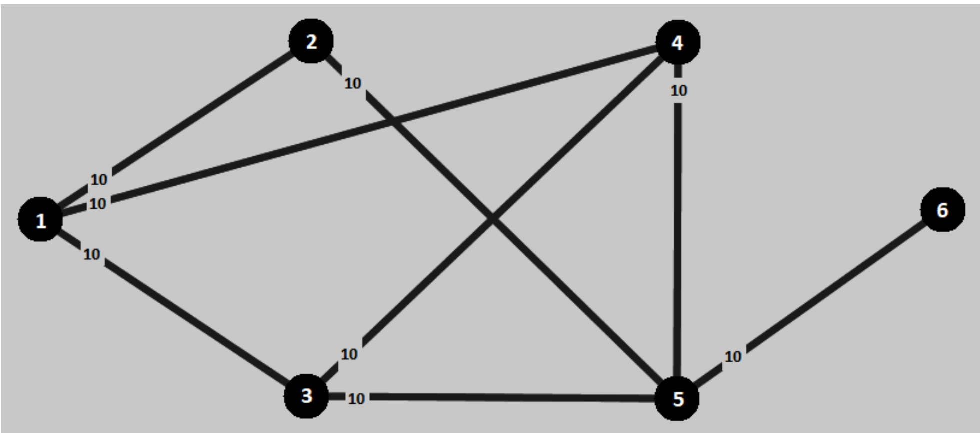

Figura 1.24: ODU2 logical topology defined by the ODU2 traffic matrix.

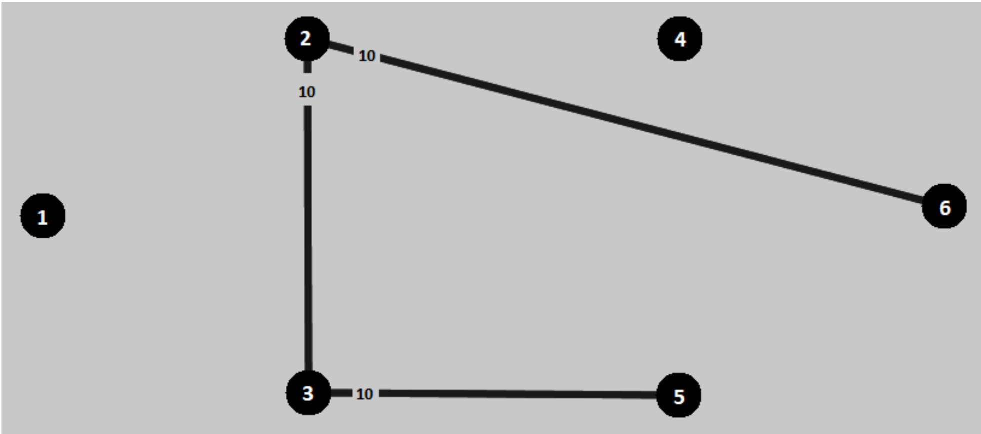

Figura 1.25: ODU3 logical topology defined by the ODU3 traffic matrix.

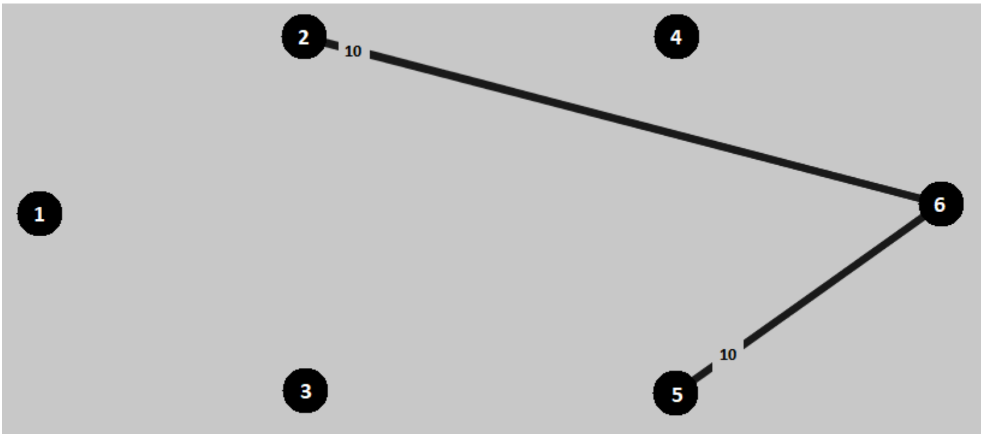

Figura 1.26: ODU4 logical topology defined by the ODU4 traffic matrix.

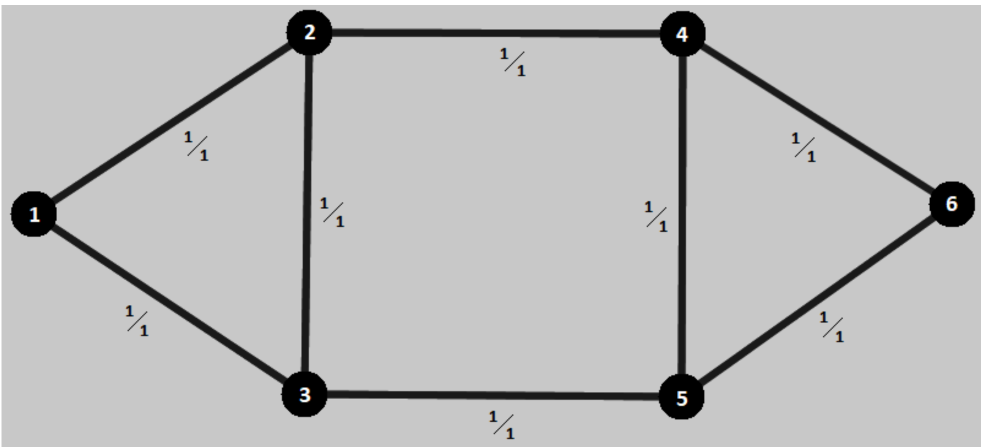

Figura 1.27: Physical topology after dimensioning.

Following all the steps mentioned in the **??**, applying the routing and grooming heuristic algorithms in the Net2Plan software and using all the data referring to this scenario, the obtained result for the Vasco's heuristics can be consulted in the following table 1.6. In table 1.5 mentioned in previous scenario we can see how all the values were calculated.

<table>
<tr><th colspan="7">CAPEX of the Network</th></tr>
<tr><th colspan="3"></th><th>Quantity</th><th>Unit Price</th><th>Cost</th><th>Total</th></tr>
<tr><td rowspan="3">Link Cost</td><td colspan="2">OLTs</td><td>16</td><td>15 000 €</td><td>240 000 €</td><td rowspan="3">77 020 000 €</td></tr>
<tr><td colspan="2">100 Gbits/s Transceivers</td><td>153</td><td>5 000 €/Gbit/s</td><td>76 500 000 €</td></tr>
<tr><td colspan="2">Amplifiers</td><td>70</td><td>4 000 €</td><td>280 000 €</td></tr>
<tr><td rowspan="9">Node Cost</td><td rowspan="7">Electrical</td><td>EXCs</td><td>6</td><td>10 000 €</td><td>60 000 €</td><td rowspan="9">15 385 900 €</td></tr>
<tr><td>ODU0 Ports</td><td>600</td><td>10 €/port</td><td>6 000 €</td></tr>
<tr><td>ODU1 Ports</td><td>500</td><td>15 €/port</td><td>7 500 €</td></tr>
<tr><td>ODU2 Ports</td><td>160</td><td>30 €/port</td><td>4 800 €</td></tr>
<tr><td>ODU3 Ports</td><td>60</td><td>60 €/port</td><td>3 600 €</td></tr>
<tr><td>ODU4 Ports</td><td>40</td><td>100 €/port</td><td>4 000 €</td></tr>
<tr><td>Line Ports</td><td>153</td><td>100 000 €/port</td><td>15 300 000 €</td></tr>
<tr><td rowspan="2">Optical</td><td>OXCs</td><td>0</td><td>20 000 €</td><td>0 €</td></tr>
<tr><td>Ports</td><td>0</td><td>2 500 €/port</td><td>0 €</td></tr>
<tr><td colspan="6">Total Network Cost</td><td>92 405 900 €</td></tr>
</table>

Tabela 1.6: Table with detailed description of CAPEX of Vasco's 2016 results.

High Traffic Scenario:

In this scenario we have to take into account the traffic calculated in **??**. In a first phase we will show the various existing topologies of the network. The first are the allowed topologies, physical and optical topologies, the second are the logical topology for all ODUs and finally the resulting physical topology.

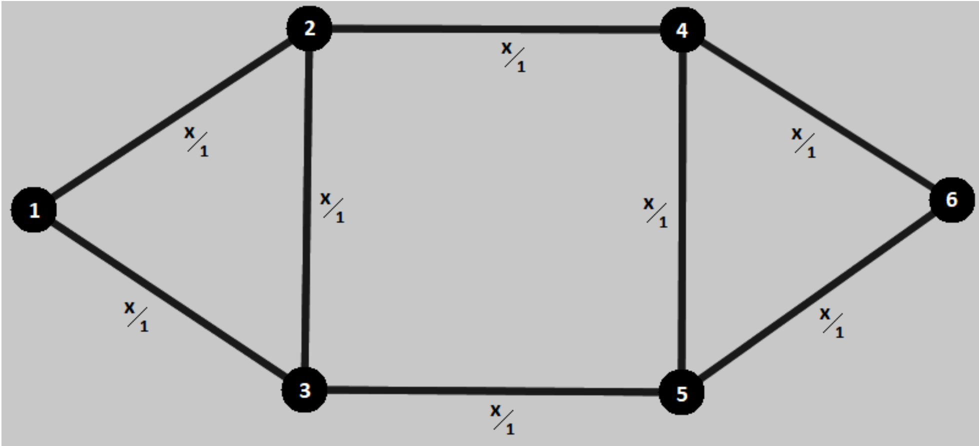

Figura 1.28: Allowed physical topology. The allowed physical topology is defined by the duct and sites in the field. It is assumed that each duct supports up to 1 bidirectional transmission system and each site supports up to 1 node.

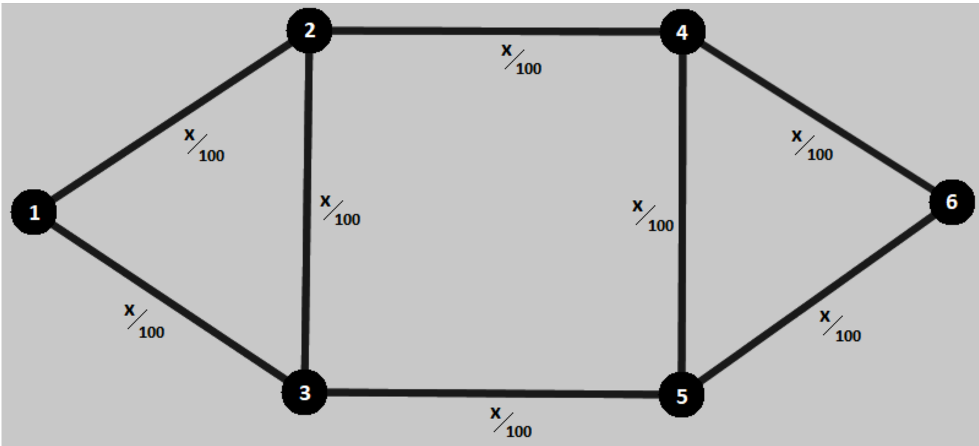

Figura 1.29: Allowed optical topology. The allowed optical topology is defined by the transport mode (opaque transport mode in this case). It is assumed that each transmission system supports up to 100 optical channels.

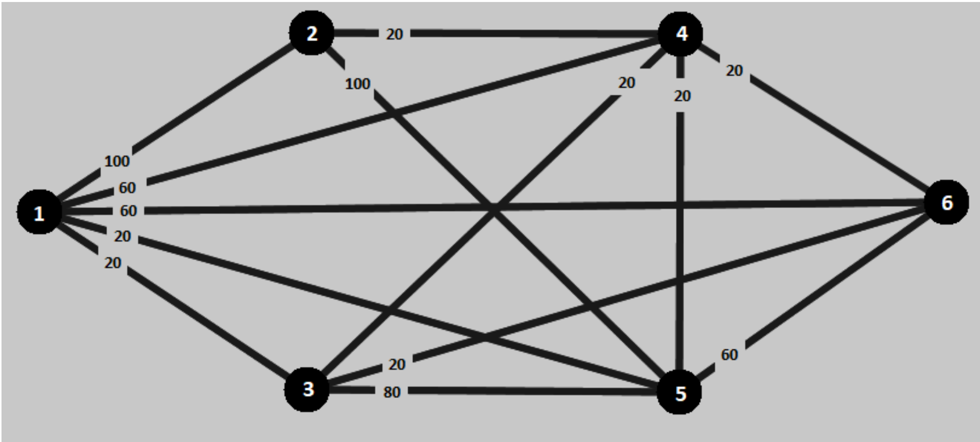

Figura 1.30: ODU0 logical topology defined by the ODU0 traffic matrix.

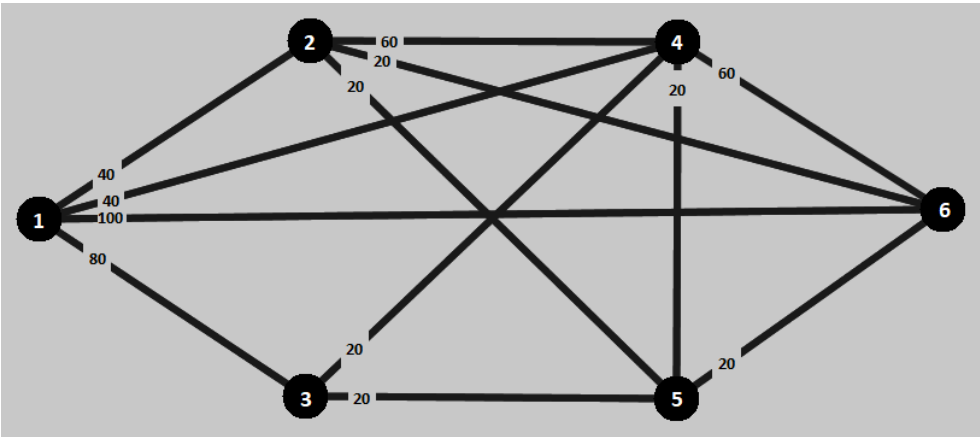

Figura 1.31: ODU1 logical topology defined by the ODU1 traffic matrix.

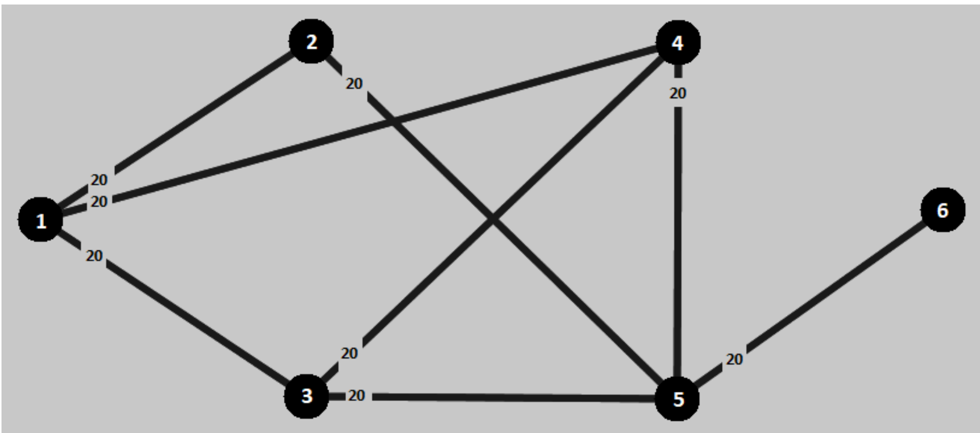

Figura 1.32: ODU2 logical topology defined by the ODU2 traffic matrix.

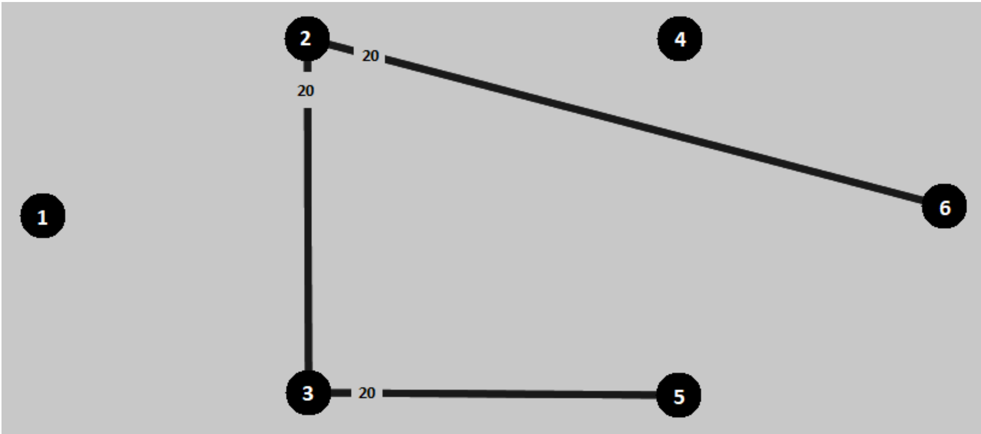

Figura 1.33: ODU3 logical topology defined by the ODU3 traffic matrix.

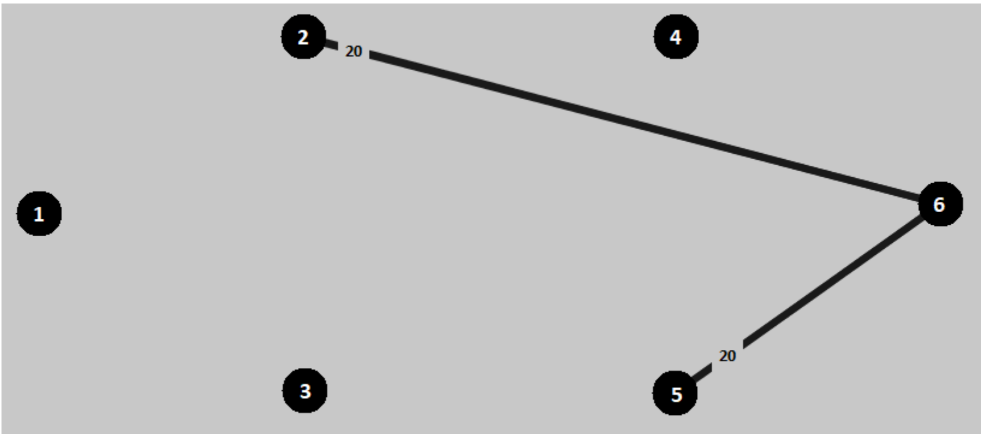

Figura 1.34: ODU4 logical topology defined by the ODU4 traffic matrix.

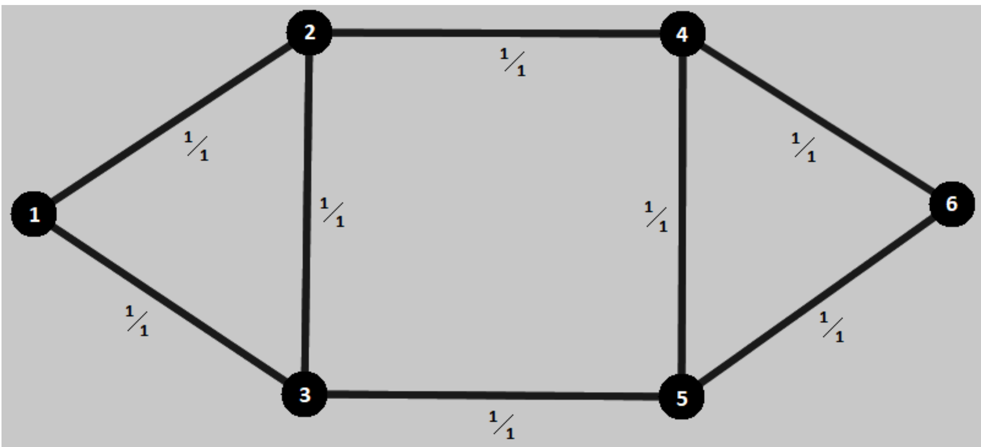

Figura 1.35: Physical topology after dimensioning.

Following all the steps mentioned in the **??**, applying the routing and grooming heuristic algorithms in the Net2Plan software and using all the data referring to this scenario, the obtained result for the Vasco's heuristics can be consulted in the following table 1.7. In table 1.5 mentioned in previous scenario we can see how all the values were calculated.

CAPEX of the Network						
			Quantity	Unit Price	Cost	Total
Link Cost	OLTs		16	15 000 €	240 000 €	149 020 000 €
	100 Gbits/s Transceivers		297	5 000 €/Gbit/s	148 500 000 €	
	Amplifiers		70	4 000 €	280 000 €	
Node Cost	Electrical	EXCs	6	10 000 €	60 000 €	29 811 800 €
		ODU0 Ports	1 200	10 €/port	12 000 €	
		ODU1 Ports	1 000	15 €/port	15 000 €	
		ODU2 Ports	320	30 €/port	9 600 €	
		ODU3 Ports	120	60 €/port	7 200 €	
		ODU4 Ports	80	100 €/port	8 000 €	
		Line Ports	297	100 000 €/port	29 700 000 €	
	Optical	OXCs	0	20 000 €	0 €	
		Ports	0	2 500 €/port	0 €	
Total Network Cost						178 831 800 €

Tabela 1.7: Table with detailed description of CAPEX of Vasco's 2016 results.

Conclusions

Once we have obtained the results for all the scenarios we will now draw some conclusions about these results. For a better analysis of the results will be created the table 1.8 with the number of line ports, tributary ports and transceivers because they are important values for the cost of CAPEX, the cost of links, the cost of nodes and finally the cost of CAPEX.

	Low Traffic	Medium Traffic	High Traffic
Traffic (Gbit/s)	500	5 000	10 000
Bidirectional Links used	8	8	8
Number of Line ports	23	153	297
Number of Tributary ports	136	1 360	2 720
Number of Transceivers	23	153	297
Link Cost	12 020 000 €	77 020 000 €	149 020 000 €
Node Cost	2 362 590 €	15 385 900 €	29 811 800 €
CAPEX	**14 382 590 €**	**92 405 900 €**	**178 831 800 €**
CAPEX/Gbit/s	**28 765 €/Gbit/s**	**18 481 €/Gbit/s**	**17 883 €/Gbit/s**

Tabela 1.8: Table with different value of CAPEX for this case.

Looking at the previous table we can make some comparisons between the several scenario:

- Comparing the low traffic with the others we can see that despite having an increase of factor ten (medium traffic) and factor twenty (high traffic), the same increase does not occur in the final cost (it is lower);

  This happens because the number of the transceivers is lower than expected which leads by carrying the traffic with less network components and, consequently, the network CAPEX is lower;

- Comparing the medium traffic with the high traffic we can see that the increase of the factor is double and in the final cost this factor is very close but still inferior;

  This happens because the number of the transceivers is also lower but very close to the expected;

- Comparing the CAPEX cost per bit we can see that in the low traffic the cost is higher than the medium and high traffic, which in these two cases the value is very similar;

  This happens because the lower the traffic, the higher CAPEX/bit will be. We can see that in medium and high traffic the results tend to be one closer value.

Opens Issues

The creation of this model for any scenario, started with some considerations and some open issues being:

- Allow blocking.

  The presented model assume that the solution is possible or impossible, does not support a partial solution where some demands are not routed (are blocked);

- Allow multiple transmission system.

  The presented model for each link only supports one transmission system;

- Allowing multi-path routing.

  The presented model for all demands sharing the same node pairs have to follow the same path.

Allow Blocking

Student Name	:	Élio Coelho (08/10/2018 -)
Goal	:	Allows blocking, i.e., some demands are not routed.

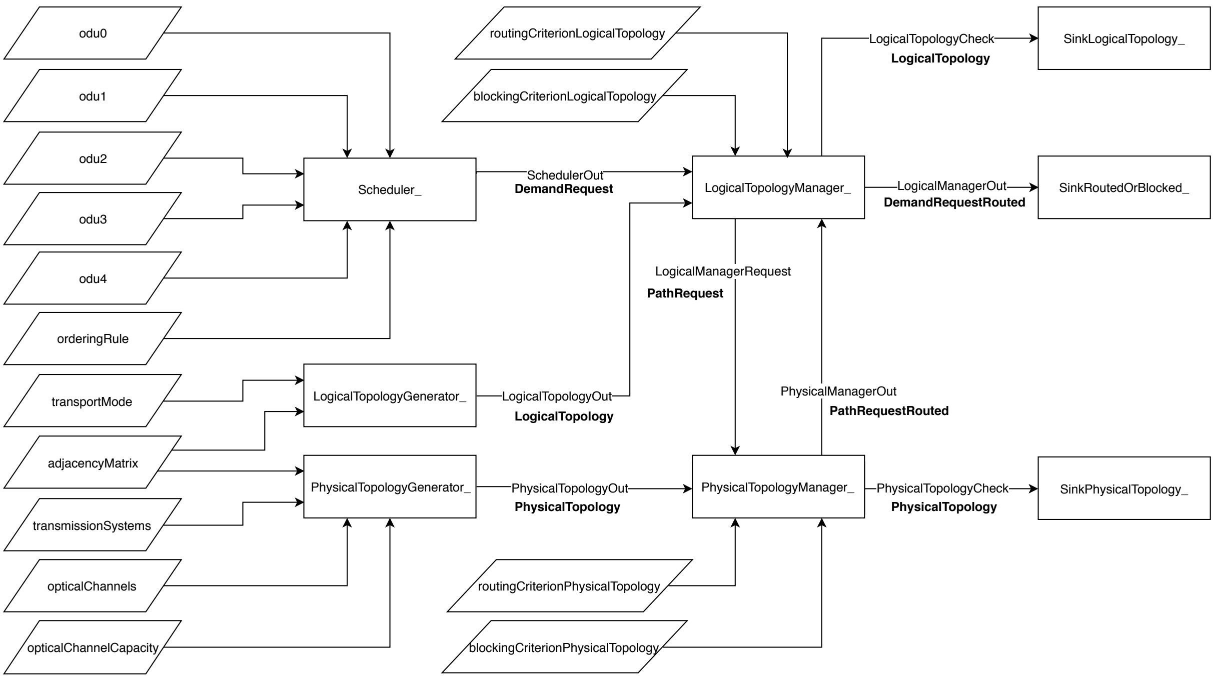

Figura 1.36: Simulation Diagram

Input Parameter	Default Value	Type	Description
odu0	[0]	matrix	ODU0 demands matrix
odu1	[0]	matrix	ODU1 demands matrix
odu2	[0]	matrix	ODU2 demands matrix
odu3	[0]	matrix	ODU3 demands matrix
odu4	[0]	matrix	ODU4 demands matrix
orderingRule	0	integer	Demands ordering rule: 0 - ODU4 to ODU0 1 - ODU0 to ODU4
transportMode	opaque	string	Transport mode: opaque transparent translucent
adjacencyMatrix	[0]	matrix	Adjacency matrix of the physical network
transmissionSystems	1	integer	Number of transmission systems
opticalChannels	100	integer	Number of optical channels per link
opticalChannelCapacity	80	integer	Capacity of each optical channel in ODU0s
routingCriterionLogicalTopology	hops	string	Shortest path type: hops km
blockingCriterionLogicalTopology	3	integer	Number of short paths created between each pair of nodes
routingCriterionPhysicalTopology	hops	string	Shortest path type: hops km
blockingCriterionPhysicalTopology	3	integer	Number of short paths created between each pair of nodes

Tabela 1.9: System Input Parameters

Signal Name	Signal Type
SchedulerOut	DemandRequest
LogicalTopologyOut	LogicalTopology
PhysicalTopologyOut	PhysicalTopology
LogicalManagerRequest	PathRequest
PhysicalManagerOut	PathRequestRouted
LogicalManagerOut	DemandRequestRouted
LogicalTopologyCheck	LogicalTopology
PhysicalTopologyCheck	PhysicalTopology

Tabela 1.10: System Signals

DemandRequest

demandIndex	sourceNode	destinationNode	oduType	survivabilityMethod
0...D-1	1...N	1...N	0...4	0 - None 1 - Protection 1+1

Tabela 1.11: DemandRequest variable

D = numberOfDemands
N = numberOfNodes

LogicalTopology

matrix NxN logicalTopology
vector paths

Node	1	...	...	N
1	0	X	X	X
...	X	0	X	X
...	X	X	0	X
N	X	X	X	0

Tabela 1.12: logicalTopology matrix

N = numberOfNodes
X = 0 => not logical link or X = 1 => logical link

path

pathIndex	**sourceNode**	**intermediateNodes**	**destinationNode**	**capacity** (ODU0s)	**numberOfLightPaths**
0...P-1	1...N	-1 or 1...N	1...N	0...OC	1...LP

Tabela 1.13: path variable

P = numberOfPaths
N = numberOfNodes
OC = opticalChannelCapacity
LP = numberOfLightPaths

lightPath

lightPathIndex	**sourceNode**	**destinationNode**	**capacity** (ODUOs)	numberOfOpticalChannels	**optica**
0...LP-1	1...N	1...N	1...OC	1...Och	vector<op

Tabela 1.14: lightPath variable

LP = numberOfLightPaths
N = numberOfNodes
Och = numberOfOpticalChannels
OC = opticalChannelCapacity

opticalChannel

opticalChannelIndex	**sourceNode**	**destinationNode**	**capacity** (ODUOs)	**demandsIndex**	**wavelength**
0...Och-1	1...N	1...N	1...OC	0...D	1...W

Tabela 1.15: opticalChannel variable

Och = numberOfOpticalChannels
N = numberOfNodes
OC = opticalChannelCapacity
D = numberOfDemands
W = numberOfWavelenghts

PhysicalTopology

matrix NxN physicalTopology
vector OMS

Node	1	...	...	N
1	0	X	X	X
...	X	0	X	X
...	X	X	0	X
N	X	X	X	0

Tabela 1.16: physicalTopology matrix

N = numberOfNodes
X = 0 => not physical link or X = 1 => physical link

OMS

OMSIndex	**sourceNode**	**destinationNode**	**capacity** (opticalChannels)	**wavelengths**
0	1...N	1...N	0...OchL	0 or 1...W
...	1...N	1...N	0...OchL	0 or 1...W
L-1	1...N	1...N	0...OchL	0 or 1...W

Tabela 1.17: OMS (Optical Multiplex Section) variable

L = numberOfLinks
N = numberOfNodes
OchL = numberOfOpticalChannelsPerLink
W = numberOfWavelengths

PathRequest

requestIndex	**sourceNode**	**intermediateNodes**	**destinationNode**	**wavelengthContinuity**
0...R-1	0...N	-1 or 1...N	1...N	0

Tabela 1.18: PathRequest variable

N = numberOfNodes
R = numberOfRequests

PathRequestRouted

requestIndex	**routed**	**numberOfLightPaths**	**lightPaths**
0...R-1	0 or 1	1...LP	vector <lighPaths>

Tabela 1.19: PathRequestRouted variable

R = numberOfRequests

LP = numberOfLightPaths
Och = numberOfOpticalChannels

DemandRequestRouted

demandIndex	routed	wavelength
0...D-1	0 or 1	1...W

Tabela 1.20: DemandRequestRouted variable

D = numberOfDemands
W = numberOfWavelengths

Block	Description
Scheduler_	
LogicalTopologyGenerator_	
PhysicalTopologyGenerator_	
LogicalTopologyManager_	
PhysicalTopologyManager_	
SinkRoutedOrBlocked_	
SinkLogicalTopology_	
SinkPhysicalTopology_	

Tabela 1.21: System Blocks

Input Parameters	State Variables	Output Signal
odu1 odu2 odu3 odu4 orderingRule	odu1 odu2 odu3 odu4 demandIndex numberOfDemands	SchedulerOut

Tabela 1.22: Scheduler_ Block

Input Parameters	State Variables	Output Signal
transportMode adjacencyMatrix		LogicalTopologyOut

Tabela 1.23: LogicalTopologyGenerator_ Block

Input Parameters	State Variables	Output Signal
adjacencyMatrix transmissionSystems opticalChannels opticalChannelCapacity		PhysicalTopologyOut

Tabela 1.24: PhysicalTopologyGenerator_ Block

Input Parameters	Input Signals	State Variables	Output Signals
routingCriterionLogicalTopology blockingCriterionLogicalTopology	SchedulerOut LogicalTopologyOut PhysicalManagerOut		LogicalManagerRequest LogicalManagerOut LogicalTopologyCheck

Tabela 1.25: LogicalTopologyManager_ Block

Input Parameters	Input Signals	State Variables	Output Signals
routingCriterionPhysicalTopology	LogicalManagerRequest		PhysicalManagerOu
blockingCriterionPhysicalTopology	PhysicalTopologyOut		PhysicalTopologyChe

Tabela 1.26: PhysicalTopologyManager_ Block

Input Signal
LogicalTopologyCheck

Tabela 1.27: SinkLogicalTopology_ Block

Input Signal
LogicalManagerOut

Tabela 1.28: SinkRoutedOrBlocked_ Block

Input Signal
PhysicalTopologyCheck

Tabela 1.29: SinkPhysicalTopology_ Block

Example:

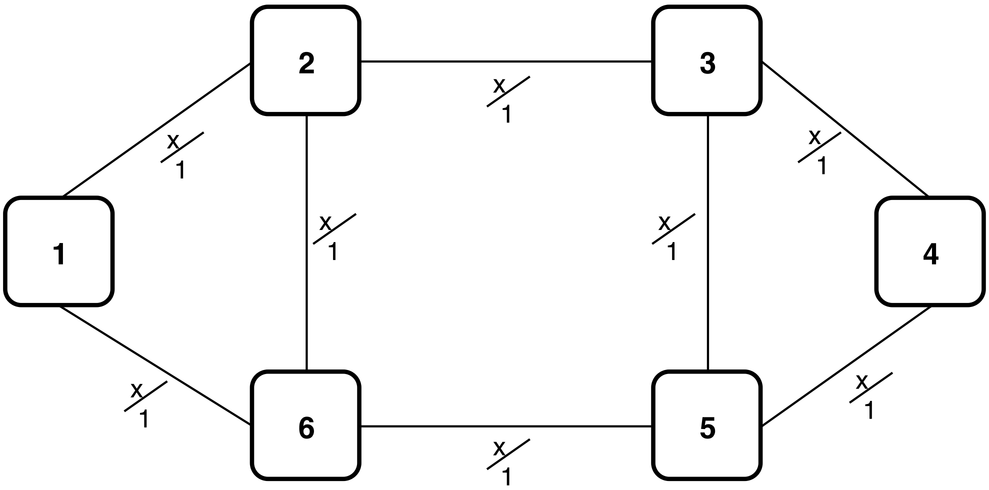

Figura 1.37: Allowed physical topology

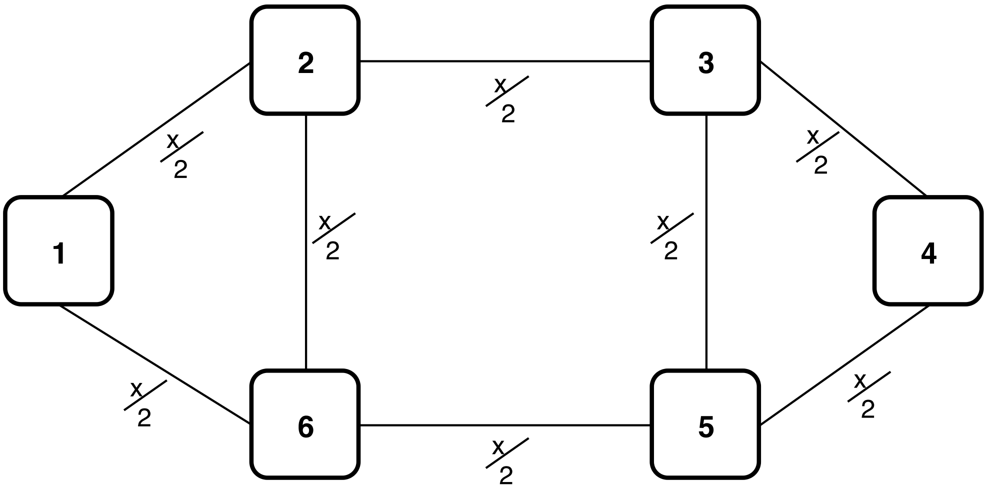

Figura 1.38: Allowed optical topology. It is assumed that each transmission system supports up to 2 optical channels.

The traffic matrices:

$$ODU0 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad ODU1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$ODU2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \end{bmatrix} \quad ODU3 = \begin{bmatrix} 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$ODU4 = \begin{bmatrix} 0 & 1 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

DemandRequest

demandIndex	**sourceNode**	**destinationNode**	**oduType**	**survivabilityMethod**
0	1	2	4	0

Tabela 1.30: DemandRequest

LogicalTopology

Node	1	2	3	4	5	6
1	0	1	0	0	0	1
2	1	0	1	0	0	1
3	0	1	0	1	1	0
4	0	0	1	0	1	0
5	0	0	1	1	0	1
6	1	1	0	0	1	0

Tabela 1.31: logicalTopology matrix

paths

pathIndex	**sourceNode**	**intermediateNodes**	**destinationNode**	**capacity** (ODU0s)	**numberOfLightPaths**

Tabela 1.32: paths variable

PhysicalTopology

Node	1	2	3	4	5	6
1	0	1	0	0	0	1
2	1	0	1	0	0	1
3	0	1	0	1	1	0
4	0	0	1	0	1	0
5	0	0	1	1	0	1
6	1	1	0	0	1	0

Tabela 1.33: physicalTopology matrix

OMS

OMSIndex	**sourceNode**	**destinationNode**	**capacity** (opticalChannels)	**wavelengths**
0	1	2	2	0
1	1	6	2	0
2	2	1	2	0
3	2	3	2	0
4	2	6	2	0
5	3	2	2	0
6	3	4	2	0
7	3	5	2	0
8	4	3	2	0
9	4	5	2	0
10	5	3	2	0
11	5	4	2	0
12	5	6	2	0
13	6	1	2	0
14	6	2	2	0
15	6	5	2	0

Tabela 1.34: OMS (Optical Multiplex Section) variable

PathRequest

requestIndex	**sourceNode**	**intermediateNodes**	**destinationNode**	**wavelengthContinuity**
0	1	-1	2	0

Tabela 1.35: PathRequest variable

PathRequestRouted

requestIndex	**routed**	**numberOfLightPaths**	**lightPaths**
0	1	1	vector <lightPaths>

Tabela 1.36: PathRequestRouted variable

lightPath

lightPathIndex	**sourceNode**	**destinationNode**	**capacity** (ODU0s)	**numberOfOpticalChannels**	**optic**
0	1	2	0	1	vector<o

Tabela 1.37: lightPaths variable

opticalChannels

opticalChannelIndex	**sourceNode**	**destinationNode**	**capacity** (ODU0s)	**demandsIndex**	**wavelength**
0	1	2	0	0	1

Tabela 1.38: opticalChannels variable

OMS

OMSIndex	**sourceNode**	**destinationNode**	**capacity** (opticalChannels)	**wavelengths**
0	1	2	0	[1]
1	1	6	2	[0]
2	2	1	2	[0]
3	2	3	1	[0]
4	2	6	2	[0]
5	3	2	2	[0]
6	3	4	2	[0]
7	3	5	2	[0]
8	4	3	2	[0]
9	4	5	2	[0]
10	5	3	2	[0]
11	5	4	2	[0]
12	5	6	2	[0]
13	6	1	2	[0]
14	6	2	2	[0]
15	6	5	2	[0]

Tabela 1.39: OMS (Optical Multiplex Section) variable

paths

pathIndex	sourceNode	intermediateNodes	destinationNode	capacity (ODU0s)	numberOfLightPaths
0	1	-1	2	0	1

Tabela 1.40: path variable

DemandRequestRouted

demandIndex	routed	wavelength
0	1	1

Tabela 1.41: DemandRequestRouted variable

DemandRequest

demandIndex	sourceNode	destinationNode	oduType	survivabilityMethod
1	1	3	4	0

Tabela 1.42: DemandRequest

PathRequest

requestIndex	sourceNode	intermediateNodes	destinationNode	wavelengthContinuity
1	1	2	3	0

Tabela 1.43: PathRequest variable

PathRequestRouted

requestIndex	routed	numberOfLightPaths	lightPaths
1	1	2	vector <lightPaths>

Tabela 1.44: PathRequestRouted variable

lightPaths

lightPathIndex	sourceNode	destinationNode	capacity (ODU0s)	numberOfOpticalChannels	opti
1	1	2	0	1	vector<o
2	2	3	0	1	vector<o

Tabela 1.45: lightPaths variable

opticalChannels

opticalChannelIndex	sourceNode	destinationNode	capacity (ODU0s)	demandsIndex	wavelength
1	1	2	0	1	2
2	2	3	0	1	3

Tabela 1.46: opticalChannels variable

OMS

OMSIndex	sourceNode	destinationNode	capacity (opticalChannels)	wavelengths
0	1	2	0	[1,2]
1	1	6	2	[0]
2	2	1	2	[0]
3	2	3	1	[3]
4	2	6	2	[0]
5	3	2	2	[0]
6	3	4	2	[0]
7	3	5	2	[0]
8	4	3	2	[0]
9	4	5	2	[0]
10	5	3	2	[0]
11	5	4	2	[0]
12	5	6	2	[0]
13	6	1	2	[0]
14	6	2	2	[0]
15	6	5	2	[0]

Tabela 1.47: OMS (Optical Multiplex Section) variable

paths

pathIndex	sourceNode	intermediateNodes	destinationNode	capacity (ODU0s)	numberOfLightPaths
0	1	-1	2	0	1
1	1	2	3	0	2

Tabela 1.48: paths variable

DemandRequestRouted

demandIndex	**routed**	**wavelengths**
1	1	[2,3]

Tabela 1.49: DemandRequestRouted variable

demandIndex	sourceNode	destinationNode	oduType	survivabilityMethod
0	1	2	4	0
1	1	3	4	0
2	1	3	4	0
3	1	3	3	0
4	1	3	3	0
5	1	3	3	0
6	6	4	2	0
7	6	4	2	0
8	6	4	2	0
9	5	3	1	0

Tabela 1.50: Demands

pathIndex	sourceNode	intermediateNodes	destinationNode	capacity (ODU0s)	lightPathsIndex
0	1	null	2	0	0
1	1	2	3	0	1, 2
2	1	6, 2	3	0	3, 4, 5
3	1	6, 5	3	8	6, 7, 8
4	6	5	4	0	7, 9
5	6	5	4	64	10, 9
6	5	null	3	14	8

Tabela 1.51: Paths

sourceNode	destinationNode	shortestPaths
1	2	1->2
1	3	1->3 1->6->2->3 1->6->5->3
6	4	6->5->4
5	3	5->3

Tabela 1.52: Dijkstra

lightPathIndex	sourceNode	destinationNode	capacity (ODU0s)	opticalChannelsIndex
0	1	2	0	0
1	1	2	0	1
2	2	3	0	2
3	1	6	0	3
4	6	2	0	4
5	2	3	0	5
6	1	6	16	6
7	6	5	0	7
8	5	3	14	8
9	5	4	64	9
10	6	5	72	10

Tabela 1.53: Light paths

opticalChannelIndex	sourceNode	destinationNode	capacity (ODU0s)	demandsIndex
0	1	2	0	0
1	1	2	0	1
2	2	3	0	1
3	1	6	0	2
4	6	2	0	2
5	2	3	0	2
6	1	6	16	3, 4
7	6	5	0	3, 4, 6, 7
8	5	3	14	3, 4, 9
9	5	4	64	6, 8
10	6	5	72	8

Tabela 1.54: Optical channels

omsIndex	**sourceNode**	**destinationNode**	**capacity** (opticalChannels)	**[opticalChannelsIndex, wavelengths]**
0	1	2	0	[0,] [1,]
1	2	3	0	[2,] [5,]
2	1	6	1	[3,] [6,]
3	6	2	1	[4,]
4	6	5	1	[7,] [10,]
5	5	3	2	[8,]
6	5	4	2	[9,]

Tabela 1.55: Optical multiplex section (OMS)

File Name	**Description**	Status
netxpto_20190130.h		
scheduler_20190122.h		
logical_topology_generator_20190216.h		
logical_topology_manager_20190322.h		
physical_topology_manager_20190322.h		
sink_20180815.h		

Tabela 1.56: Header Files

File Name	**Description**	Status
netxpto_20190130.cpp		
opaque_sdf.cpp		
scheduler_20190122.cpp		
logical_topology_generator_20190216.cpp		
logical_topology_manager_20190322.cpp		
physical_topology_manager_20190322.cpp		
sink_20180815.cpp		

Tabela 1.57: Source Files

1.1.2 Opaque with 1+1 Protection

Student Name	:	Élio Coelho (08/10/2018 -)
	:	Pedro Coelho (01/03/2018 -)
Goal	:	Implement the heuristic model for the opaque transport mode with 1 plus 1 protection.

The impact of failure in WDM (Wavelength Division Multiplexing) networks is caused by extremely high volume of traffic carried. In a high speed network like the WDM, a failure of a network element may cause failure of various optical channels that leads to large data and revenue losses, which can interrupt communication services.

In this protection scheme, the primary and backup path carry the traffic end-to-end, i.e., there is a need to have a backup path (the unaffected path) in case of a network failure. Then, the receiver will decide which one of the two incoming traffic it is going to pick, if the primary or the backup path.

Although it is the fastest protection scheme, it is also the most expensive, because it normally uses more than the double of the capacity of the primary path. This happens because the backup path is typically longer than the primary.

After the creation of the matrices and the network topology, it is necessary to apply the routing and grooming algorithms created. In the end, a report algorithm will be applied to obtain the best CAPEX result for the network in question.

Firstly, in the opaque transport mode, the optical node cost is 0 because all the ports in the network are electrical. Consequently, to calculate the nodes' cost in this transport mode it only has to be considered the electrical nodes' cost:

- $N_{OXC,n} = 0, \quad \forall$ n
- $N_{EXC,n} = 1, \quad \forall$ n that process traffic

As previously mentioned, equation 1.8 refers to the number of long-reach ports of the electrical switch with bit-rate -1 in node n, $P_{exc,-1,n}$, i.e., the number of line ports of node n which can be calculated as

$$P_{exc,-1,n} = \sum_{j=1}^{N} w_{nj} \tag{1.8}$$

where w_{nj} is the number of optical channels between node n and node j.

As previously mentioned, equation 1.9 refers to the number of short-reach ports of the electrical switch with bit-rate c in node n, $P_{exc,c,n}$, i.e., the number of tributary ports with bit-rate c in node n which can be calculated as

$$P_{exc,c,n} = \sum_{d=1}^{N} D_{nd,c} \tag{1.9}$$

where $D_{nd,c}$ are the client demands between nodes n and d with bit rate c.

In this case there is the following particularity:

- When n=j, the value of client demands is always zero, i.e, $D_{nn,c} = 0$.

To implement this heuristic approach there are used algorithms made in Java in a programming software called Eclipse and they are tested in an open-source network program called Net2Plan. In the Net2Plan guide section **??** there is an explanation on how to use and test them in this network planner.

In the next pages it will be described all the steps performed to obtain the final results in the opaque transport mode with 1+1 protection. In the figure below 1.39 it is shown a fluxogram with the description of this transport mode approach.

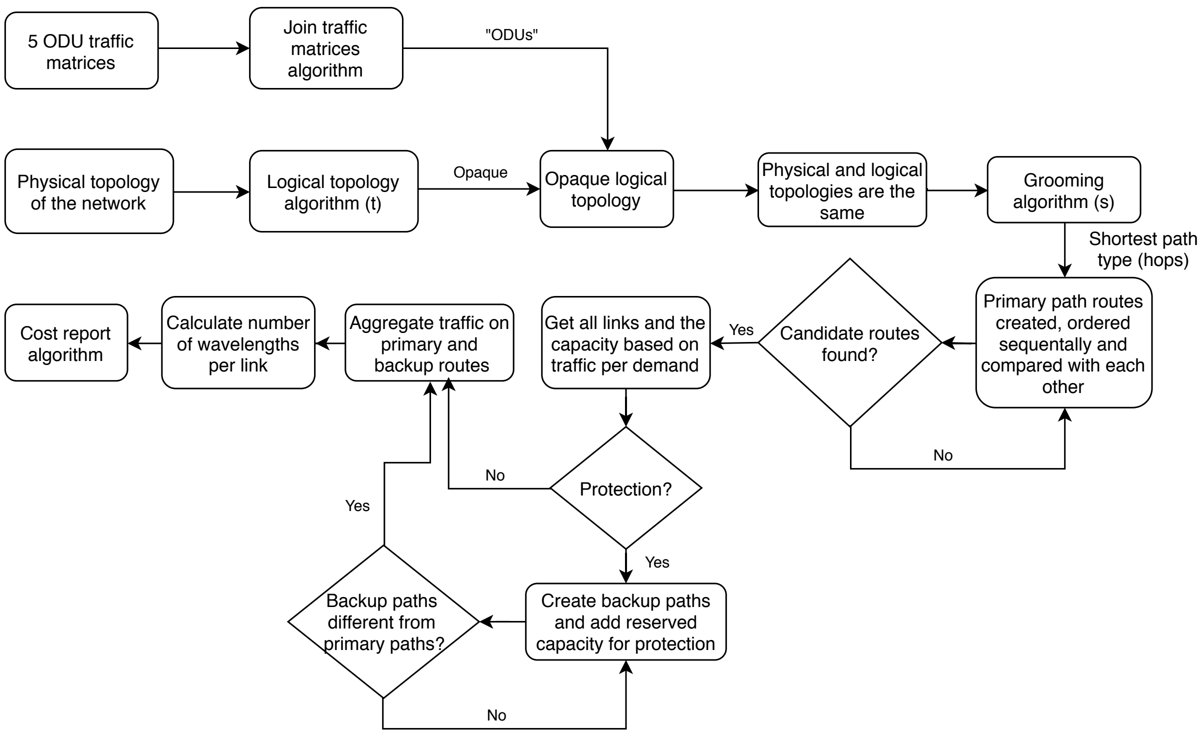

Figura 1.39: Fluxogram with the steps performed in the opaque with 1+1 protection transport mode approach.

Creation and join the traffic matrices

The first step is to create the traffic matrices based on the reference network **??**. In order to create the 5 traffic matrices in Net2Plan it is necessary the length of all the links and the total traffic used in this network, so later it is needed to define in Net2Plan the length in all end nodes and the total traffic depends on the value of traffic used (low traffic - 0.5 Tbit/s, medium traffic - 5 Tbit/s and high traffic - 10 Tbit/s). As you can see in the figure below, it is defined the path of the 5 ODUs and they will be aggregated in just one single ODU, making it possible to join all the demands in just one file and load it later into the network. This final resulting ODU joins the multiple traffic demands from all the traffic matrices previously created and, of course, the traffic demands will depend on the values used on the creation of the matrices (low, medium and high traffic).

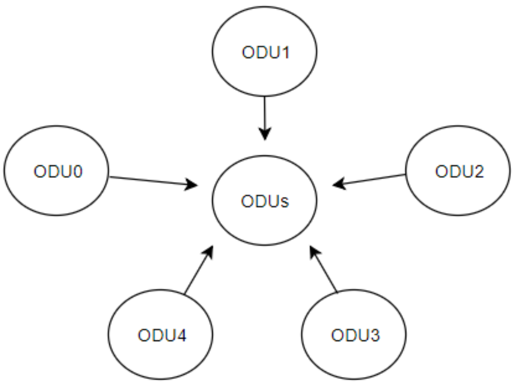

Figura 1.40: Join the 5 ODU traffic matrices into 1 single file "ODUs". The 5 traffic demands from the traffic matrices previously created are joined into 1 file to load it later on Net2Plan.

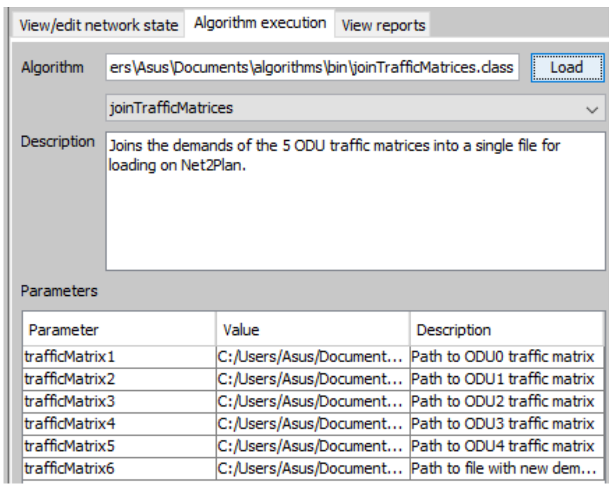

Figura 1.41: Load of the join traffic matrices algorithm for the opaque transport mode on Net2Plan. It is defined the 5 paths to load the 5 ODU traffic matrices and the last path is the one where will be saved the file that joins all 5 the traffic demands.

Creation of the physical topology

The next step is to create the allowed physical topology of the network in Net2Plan. This network consists in 6 nodes and 8 bidirectional links. It is now also possible to define the length in all links. In the figure below it is shown the allowed physical topology in this transport mode.

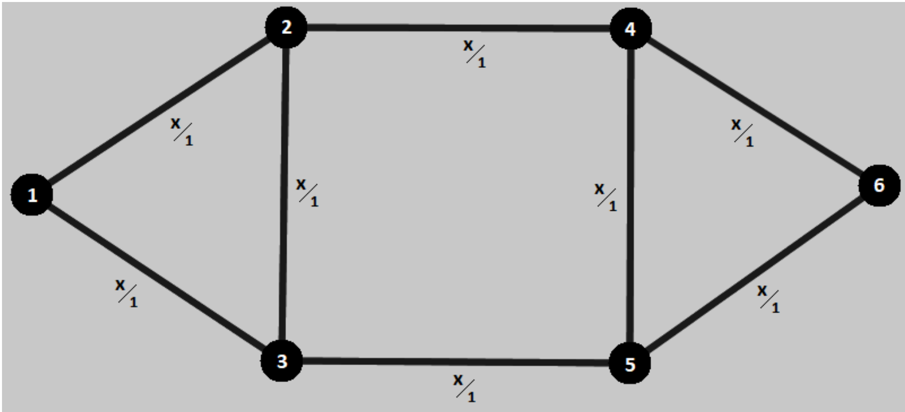

Figura 1.42: Allowed physical topology. The allowed physical topology is defined by the duct and sites in the field. It is assumed that each duct supports up to 1 bidirectional transmission system and each site supports up to 1 node.

Creation of the logical topology

It is now time to create the allowed logical topology. A network topology represents how the links and the nodes of the network interconnect with each other and the logical topology algorithm creates the logical topology on another layer. In the opaque transport mode the physical and logical topologies are the same, so it is needed to add a new layer from the lower layer (default layer). The lower layer is the physical layer of the network and it is now created a new upper layer which is the logical layer of the network and represents the logical topology of the opaque transport mode. The allowed physical and optical topologies, the logical topologies for all ODUs and the resulting physical topology is shown in the next section below 1.1.2 for the three traffic scenarios. It is shown below three figures with the code in Java of the creation of the network logical topology, the load of the logical topology algorithm in Net2Plan and the resulting allowed optical topology for the opaque transport mode with 1+1 protection.

```
if (netPlan.isSingleLayer() && logicalTopology.equalsIgnoreCase("Opaque"))
{

    lowerLayer = netPlan.getNetworkLayerDefault();
    upperLayer = netPlan.addLayerFrom(lowerLayer);
    netPlan.setRoutingType(RoutingType.HOP_BY_HOP_ROUTING, upperLayer);
    lowerLayer.setName("Physical Topology");
    upperLayer.setName("Logical Topology Opaque");
    upperLayer.setDescription("Opaque Logical Topology");

}
```

Figura 1.43: Java code of the logical topology approach for the opaque transport mode. The logical layer is created from the physical layer, as in this transport mode they are the same. The new layer is now the opaque logical topology of the network.

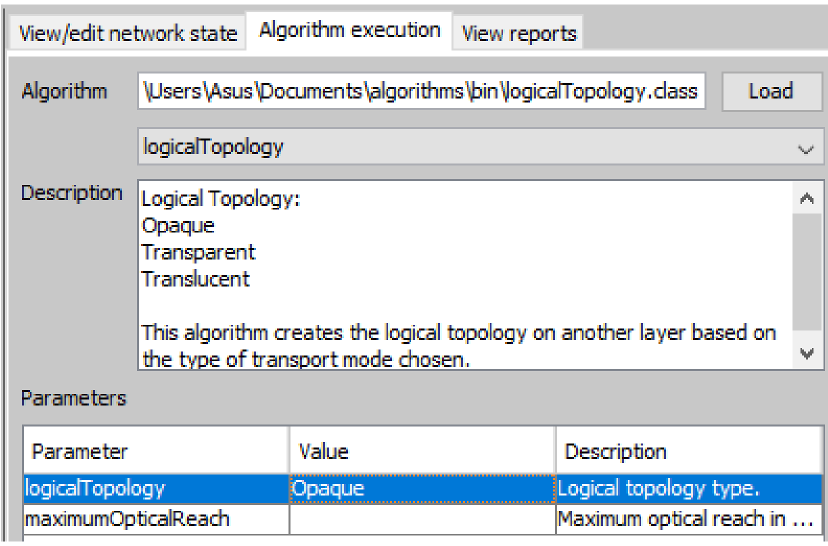

Figura 1.44: Load of the logical topology algorithm for the opaque transport mode.

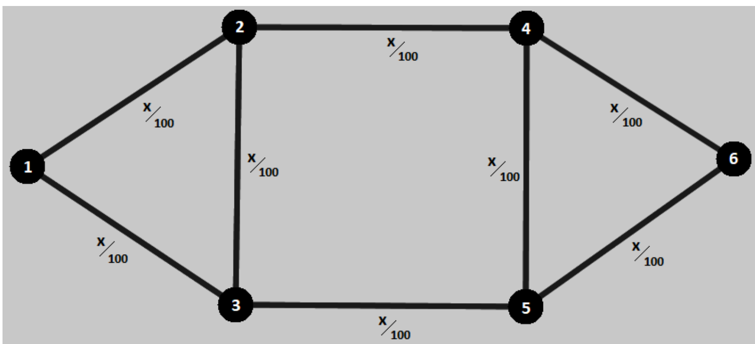

Figura 1.45: Allowed optical topology. It is assumed that each transmission system supports up to 100 optical channels.

Creation of routes and aggregation of traffic

After a network topology is created, it is now time to set the routing algorithm. In the opaque with 1+1 protection transport mode the routing algorithm starts with going through all the demands, create bidirectional routes (in this case the primary paths) based on the shortest path Dijkstra algorithm and then search the candidate routes for the respective demand. In this report it is used the shortest path type in hops. These routes are ordered sequentially and the shortest one per each demand is the primary path. The demands from the lower layer are removed and then saved in the upper layer. After this step, it is needed to set the traffic demands into the candidate routes that will integrate the network. As we also have a dedicated 1+1 protection scheme, if the network has this feature active, the algorithm will compare the previous candidate routes that will be saved to a list with the new ones that will be created. If the new routes are different from the previously created ones and if they are the next shortest path routes, then the algorithm will add these routes to the network and they will be the protection segments (backup paths) of the network. The offered traffic demands will be also set into these protection path routes. The final resulting backup path routes are used to prevent network failures. Despite of the fact the network will be much more secure, the network CAPEX will increase more than the double, due to the creation of primary and backup paths.

```
case "Logical Topology Opaque":

        for (Demand d : netPlan.getDemands(lowerLayer)) {

            boolean odd=true;
            int counter=0;

            Set<Route> droutes = d.getRoutes();
            System.out.println("droutes: " + droutes.size());

            for(Route c: droutes) {
                counter++;
                boolean jump=false;

                if(odd) {
                    c.setCarriedTraffic(d.getOfferedTraffic(), d.getOfferedTraffic());
                    save=c;
                    odd=false;
                    System.out.println("Roots");
                }
                else {
                    if (protection) {
                        List<Link>  workingpath = save.getSeqLinksRealPath();
                        System.out.println("Protection");

                        for(Link t:workingpath) {
                            if(c.getSeqLinksRealPath().contains(t)) {
                                jump=true;
                                break;
                            }
                        }
```

Figura 1.46: Creation of routes and aggregation of traffic for the opaque with 1+1 protection transport mode. The candidate routes are searched by the shortest path type and the offered traffic demands are set into these routes.

```
                        if(jump==false) {
                            ProtectionSegment segment=netPlan.addProtectionSegment(c.getSeqLinksRealPath(),
                                                                     d.getOfferedTraffic() , null);
                            save.addProtectionSegment(segment);
                            odd=true;
                            break;
                        }

                        if(jump==true && counter == droutes.size()) {
                            ProtectionSegment segment=netPlan.addProtectionSegment(c.getSeqLinksRealPath(),
                                                                     d.getOfferedTraffic() , null);
                            save.addProtectionSegment(segment);
                            odd=true;
                            throw new Net2PlanException ("Number of routes is not enough");
                        }
                    }
                }
            }
```

Figura 1.47: Creation of routes and aggregation of traffic for the opaque with 1+1 protection transport mode. The protection segments are added to all the primary paths that were chosen by the shortest path type method.

Function	Definition
netPlan.getDemands(lowerLayer)	Returns the array of demands for the lower layer.
d.getRoutes()	Returns all the routes associated to the demand "d".
c.setCarriedTraffic()	Sets the route carried traffic and the occupied capacity in the links, setting it up to be the same in all links.
d.getOfferedTraffic()	Returns the offered traffic of the demand "d".
save.getSeqLinksRealPath()	Returns the links of routes ordered sequentially.
save.addProtectionSegment(segment)	Add "segment" as a protection path in the route "save".

Tabela 1.58: Table with the description of the main functions in the creation of routes and aggregation of traffic in the grooming algorithm.

Calculation of the number of wavelengths per link

The final step of the routing and grooming algorithms is to calculate the number of wavelengths per link for the whole network. This is the last and an important step because with the number of wavelengths per link in the network, it is possible to calculate other network components. In the opaque transport mode, as in the figure below shows, the algorithm starts with going through all the links and getting the capacity based on the traffic per demand. The total carried traffic in the link including protection and non-protection segments will be divided by the wavelength capacity and it is now possible to obtain the number of wavelengths per link.

```
Link p;

for(long e:linkIds) {
    p=netPlan.getLinkFromId(e);
    double sumTraffic = p.getCarriedTrafficNotIncludingProtectionSegments() + p.getReservedCapacityForProtection();
    int nw = (int) (Math.ceil(sumTraffic/wavelengthCapacity));
    String numberWavelengths = String.valueOf(nw);
    p.setCapacity(nw*wavelengthCapacity);
    p.setAttribute("nW", numberWavelengths);
}
break;
```

Figura 1.48: Calculation of the number of wavelengths per link for the opaque transport mode. The link capacity is based on the traffic per demand.

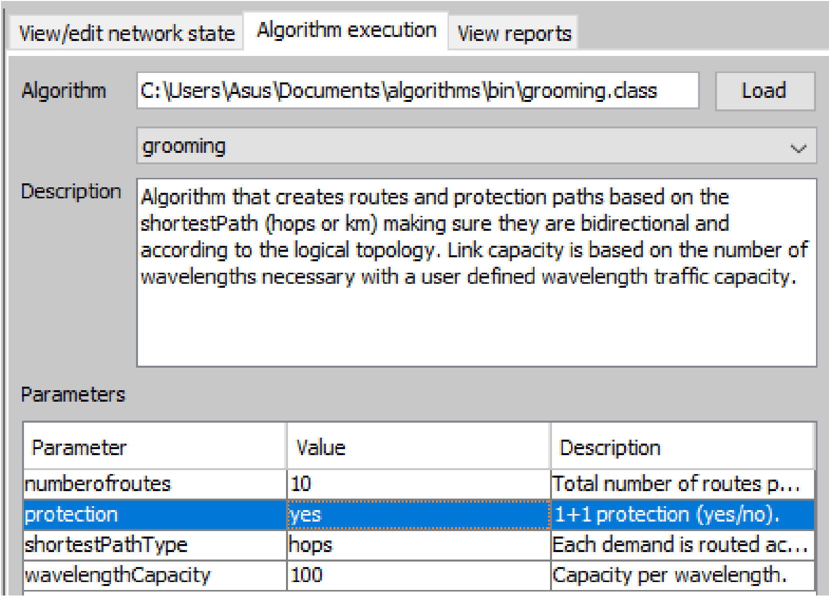

Figura 1.49: Load of the grooming algorithm for the opaque with 1+1 protection transport mode. The total number of routes per demand is set to 10, the user can define if the model is with or without protection, the shortest path type is set to "hops" and the capacity per wavelength is used 100 optical channels.

Network cost report

In order to obtain the network CAPEX results, the formulas needed to calculate the network elements and that are demonstrated previously in the beginning of this section 1.1.2 were "translated" into Java code in a cost report algorithm. This algorithm can be loaded in Net2Plan and calculates and shows in tables the network CAPEX and also the per-link and per-node information with more details. In the opaque transport mode the optical node cost is 0 because all the network ports are electrical, so it only has to be considered the electrical nodes' costs and the electrical links' costs.

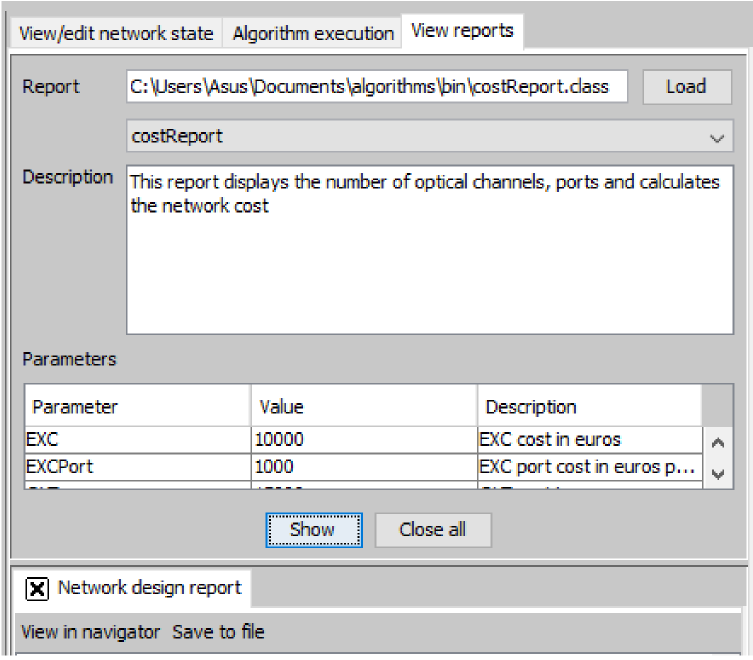

Figura 1.50: Load of the cost report algorithm on Net2Plan. The result view is an HTML page with the network optical and electrical components and their costs.

Result description

It is already known all the necessary formulas to obtain the CAPEX value for the reference network **??**. As described in the subsection of the network traffic **??**, it is necessary to obtain three different values of CAPEX for the low (0.5 Tbit/s), medium (5 Tbit/s) and high (10 Tbit/s) traffic. It is used a network software program called Net2Plan which can design the traffic matrices, create all the network topologies, simulate the algorithms into the network implemented in the programming software called Eclipse and analyze the results obtained. In this chapter will be demonstrated the results by Vasco's heuristics from 2016. In each of the three traffic scenarios, it will be shown the network topologies followed by the table with the CAPEX value of the network.

Low Traffic Scenario:

In this scenario we have to take into account the traffic calculated in **??**. In a first phase we will show the various existing topologies of the network. The first are the allowed topologies, physical and optical topologies, the second are the logical topology for all ODUs and finally the resulting physical topology.

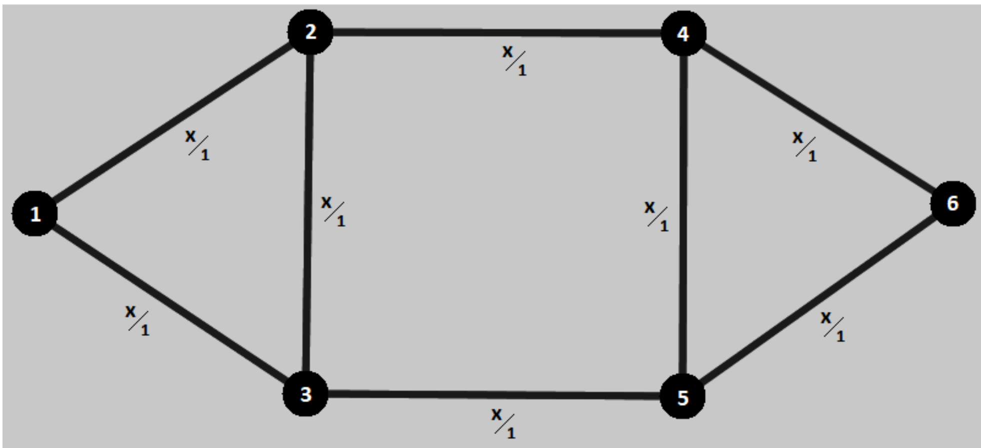

Figura 1.51: Allowed physical topology. The allowed physical topology is defined by the duct and sites in the field. It is assumed that each duct supports up to 1 bidirectional transmission system and each site supports up to 1 node.

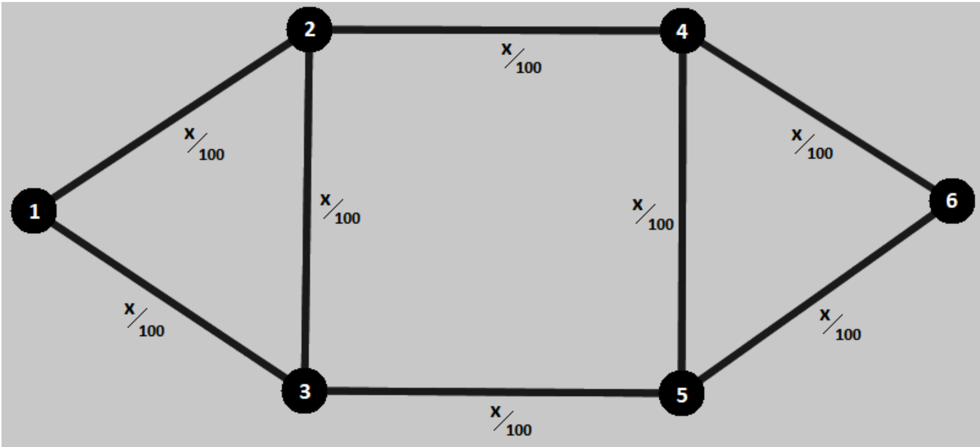

Figura 1.52: Allowed optical topology. The allowed optical topology is defined by the transport mode (opaque transport mode in this case). It is assumed that each transmission system supports up to 100 optical channels.

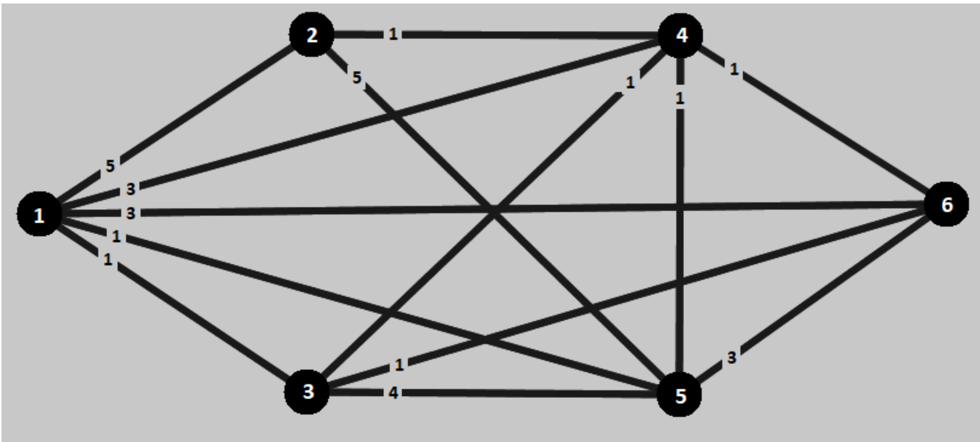

Figura 1.53: ODU0 logical topology defined by the ODU0 traffic matrix.

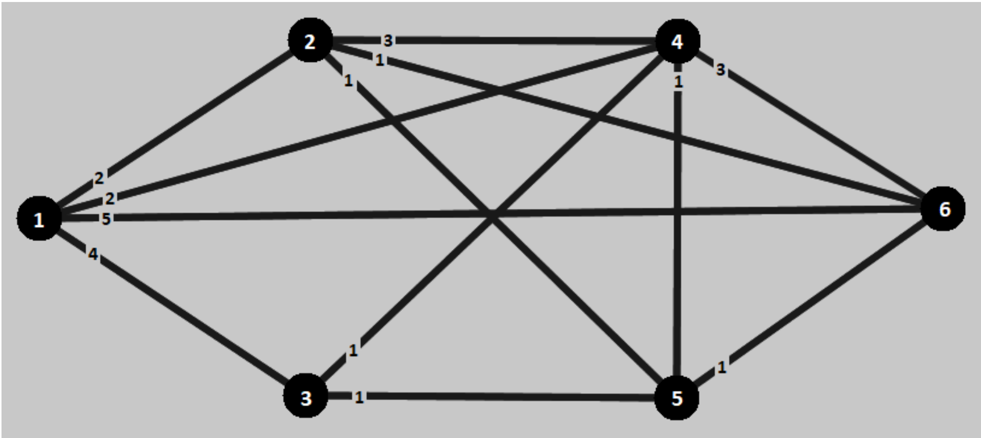

Figura 1.54: ODU1 logical topology defined by the ODU1 traffic matrix.

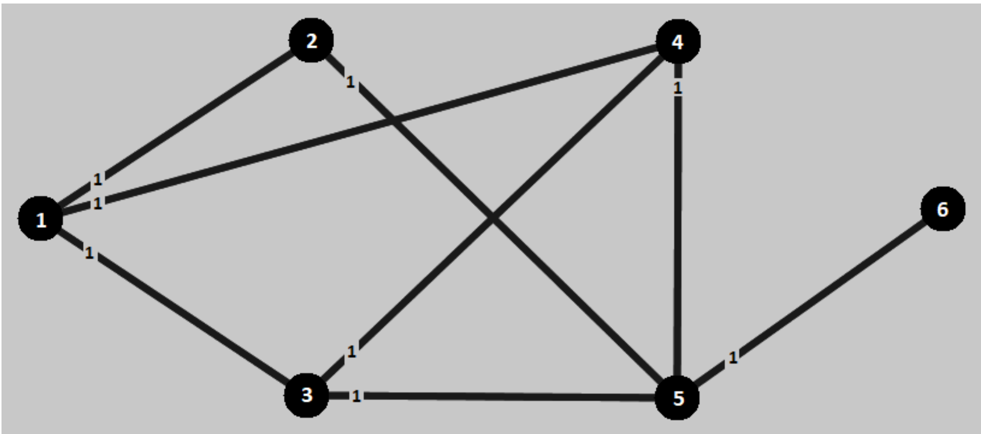

Figura 1.55: ODU2 logical topology defined by the ODU2 traffic matrix.

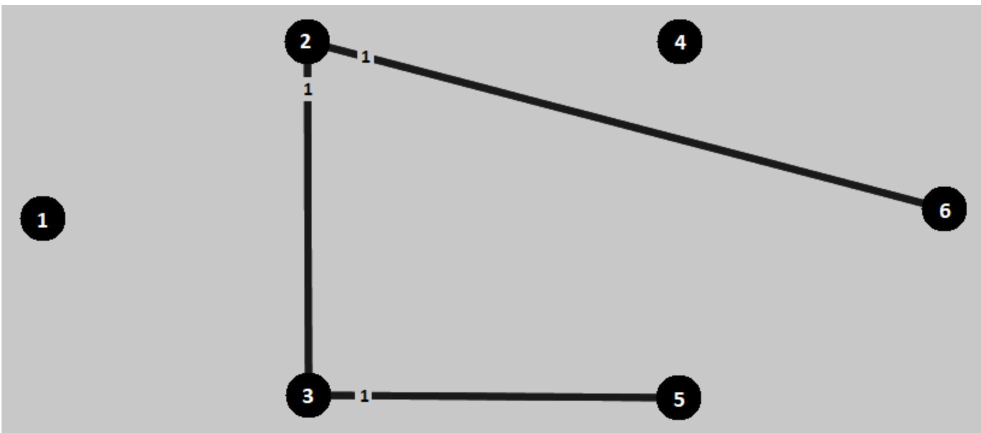

Figura 1.56: ODU3 logical topology defined by the ODU3 traffic matrix.

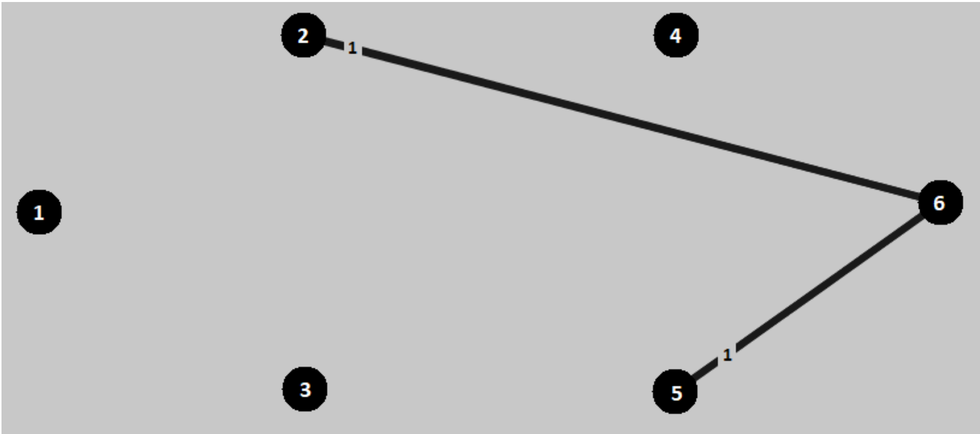

Figura 1.57: ODU4 logical topology defined by the ODU4 traffic matrix.

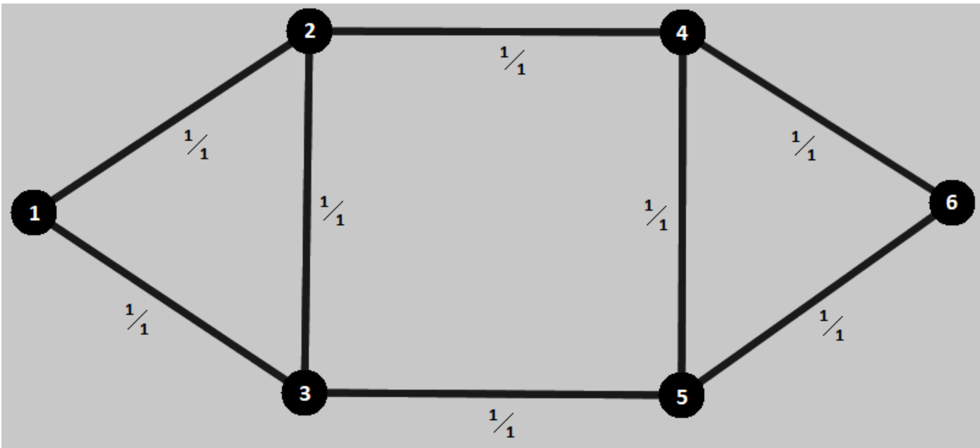

Figura 1.58: Physical topology after dimensioning.

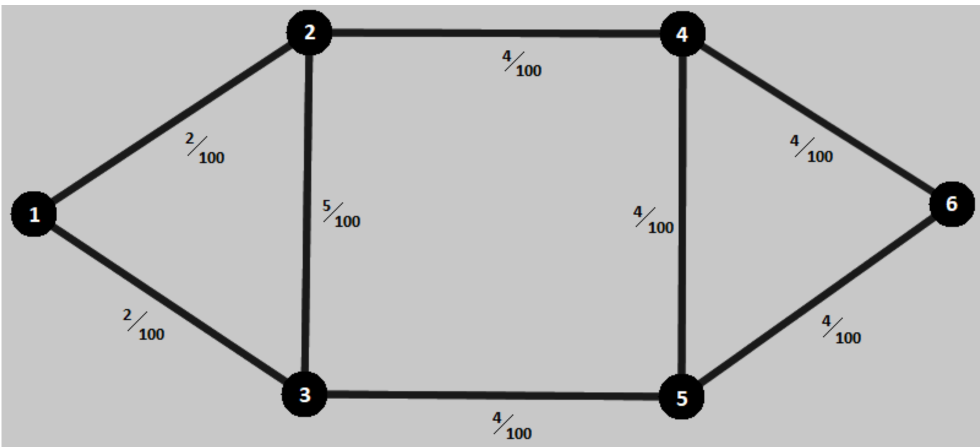

Figura 1.59: Optical topology after dimensioning.

Following all the steps mentioned in the **??**, applying the routing and grooming heuristic algorithms in the Net2Plan software and using all the data referring to this scenario, the obtained result for the Vasco's heuristics can be consulted in the following table 1.59. In table 1.5 mentioned in previous model we can see how all the values were calculated.

CAPEX of the Network						
			Quantity	Unit Price	Cost	Total
Link Cost	OLTs		16	15 000 €	240 000 €	23 520 000 €
	100 Gbits/s Transceivers		46	5 000 €/Gbit/s	23 000 000 €	
	Amplifiers		70	4 000 €	280 000 €	
Node Cost	Electrical	EXCs	6	10 000 €	60 000 €	4 662 590 €
		ODU0 Ports	60	10 €/port	600 €	
		ODU1 Ports	50	15 €/port	750 €	
		ODU2 Ports	16	30 €/port	480 €	
		ODU3 Ports	6	60 €/port	360 €	
		ODU4 Ports	4	100 €/port	400 €	
		Line Ports	46	100 000 €/port	4 600 000 €	
	Optical	OXCs	0	20 000 €	0 €	
		Ports	0	2 500 €/port	0 €	
Total Network Cost						28 182 590 €

Tabela 1.59: Table with detailed description of CAPEX of Vasco's 2016 results.

Medium Traffic Scenario:

In this scenario we have to take into account the traffic calculated in **??**. In a first phase we will show the various existing topologies of the network. The first are the allowed topologies, physical and optical topologies, the second are the logical topology for all ODUs and finally the resulting physical topology.

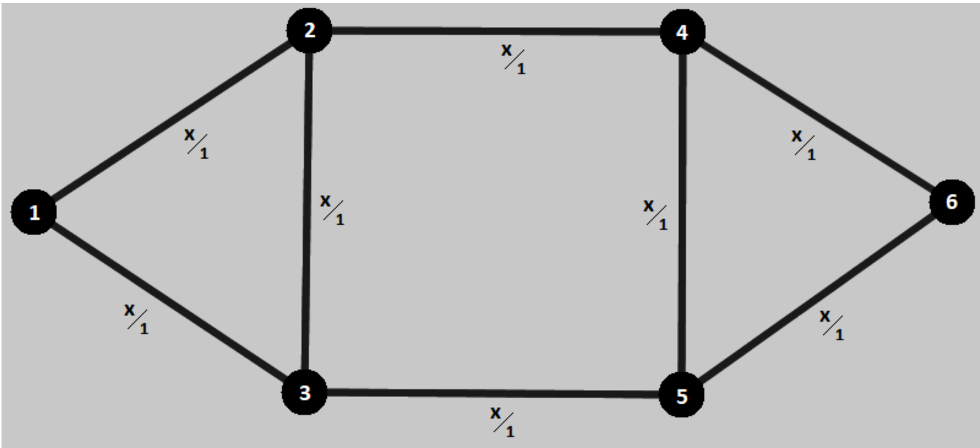

Figura 1.60: Allowed physical topology. The allowed physical topology is defined by the duct and sites in the field. It is assumed that each duct supports up to 1 bidirectional transmission system and each site supports up to 1 node.

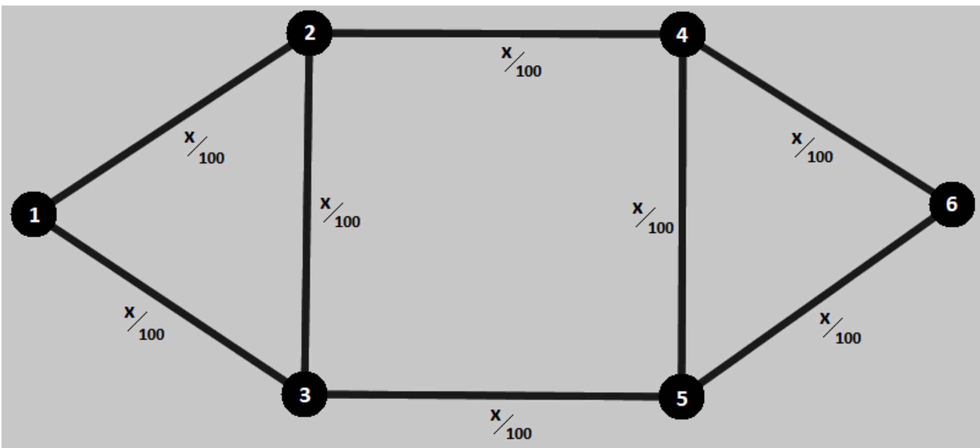

Figura 1.61: Allowed optical topology. The allowed optical topology is defined by the transport mode (opaque transport mode in this case). It is assumed that each transmission system supports up to 100 optical channels.

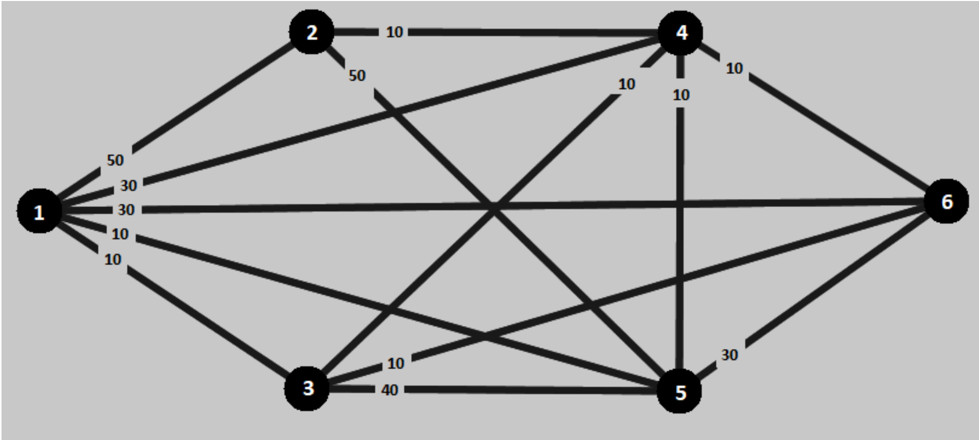

Figura 1.62: ODU0 logical topology defined by the ODU0 traffic matrix.

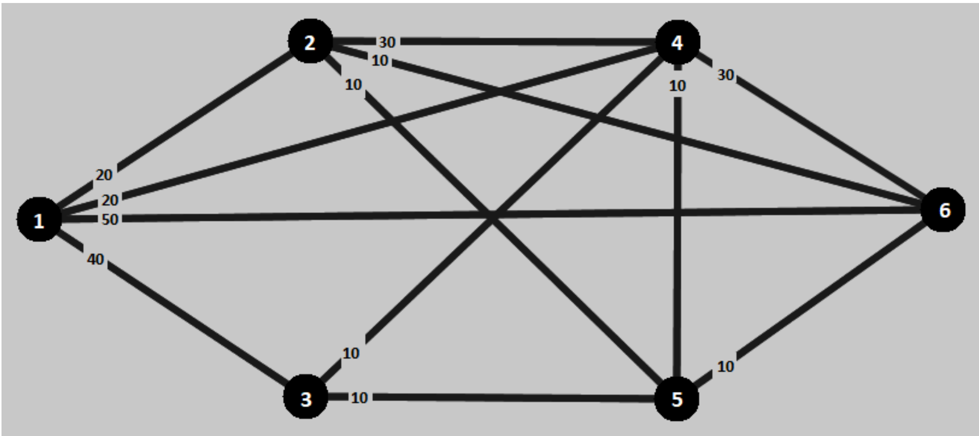

Figura 1.63: ODU1 logical topology defined by the ODU1 traffic matrix.

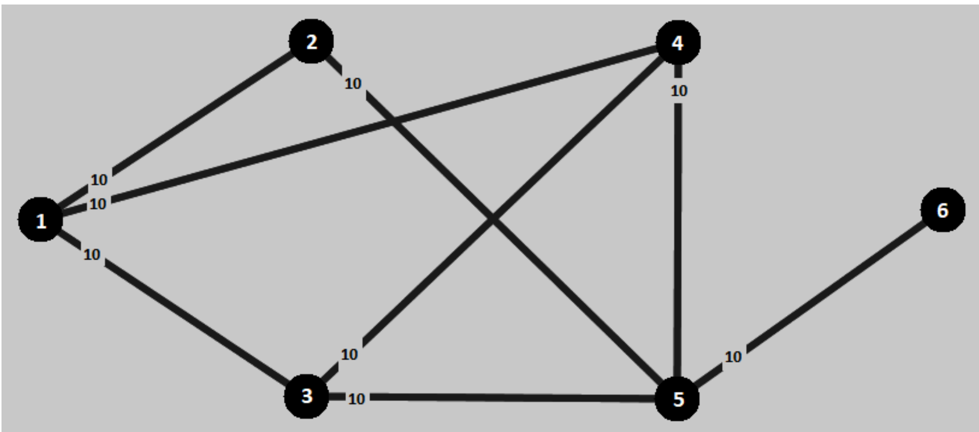

Figura 1.64: ODU2 logical topology defined by the ODU2 traffic matrix.

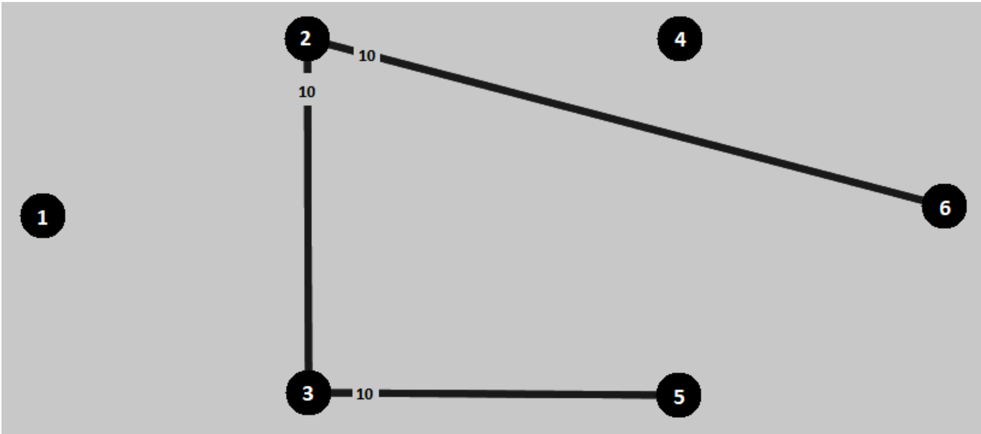

Figura 1.65: ODU03 logical topology defined by the ODU3 traffic matrix.

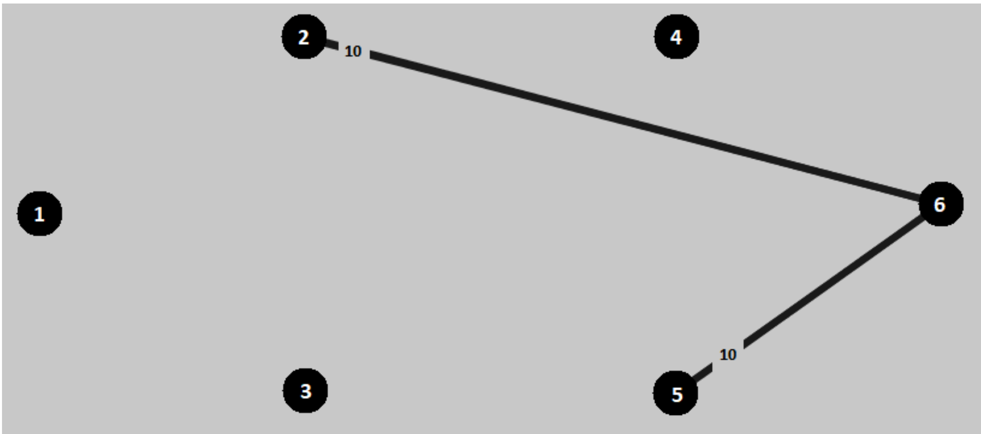

Figura 1.66: ODU4 logical topology defined by the ODU4 traffic matrix.

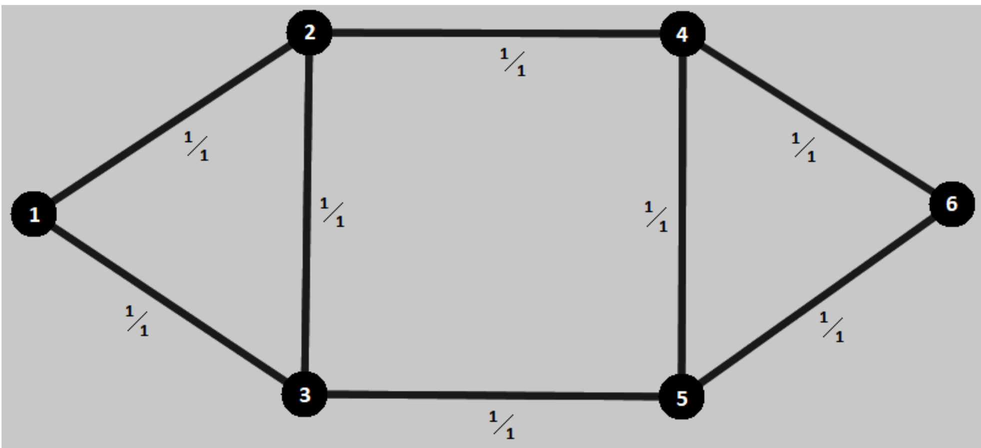

Figura 1.67: Physical topology after dimensioning.

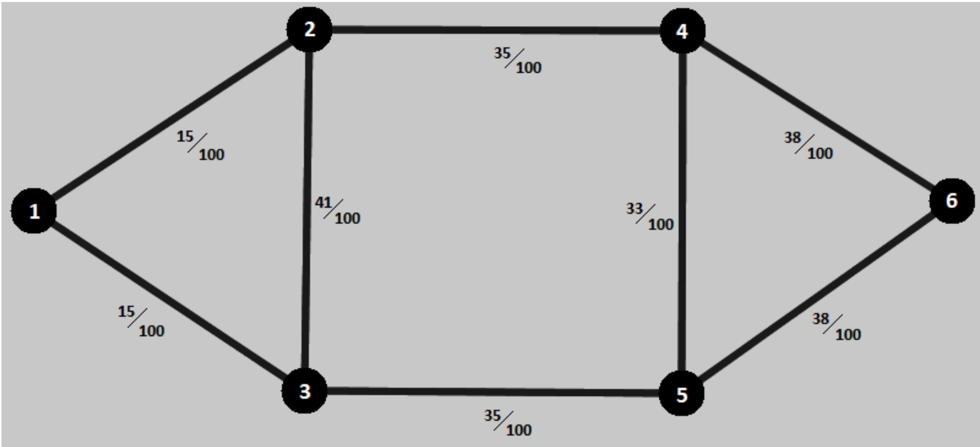

Figura 1.68: Optical topology after dimensioning.

Following all the steps mentioned in the **??**, applying the routing and grooming heuristic algorithms in the Net2Plan software and using all the data referring to this scenario, the obtained result for the Vasco's heuristics can be consulted in the following table 1.60. In table 1.5 mentioned in previous model we can see how all the values were calculated.

CAPEX of the Network						
			Quantity	Unit Price	Cost	Total
Link Cost	OLTs		16	15 000 €	240 000 €	199 520 000 €
	100 Gbits/s Transceivers		398	5 000 €/Gbit/s	199 000 000 €	
	Amplifiers		70	4 000 €	280 000 €	
Node Cost	Electrical	EXCs	6	10 000 €	60 000 €	39 885 900 €
		ODU0 Ports	600	10 €/port	6 000 €	
		ODU1 Ports	500	15 €/port	7 500 €	
		ODU2 Ports	160	30 €/port	4 800 €	
		ODU3 Ports	60	60 €/port	3 600 €	
		ODU4 Ports	40	100 €/port	4 000 €	
		Line Ports	398	100 000 €/port	39 800 000 €	
	Optical	OXCs	0	20 000 €	0 €	
		Ports	0	2 500 €/port	0 €	
Total Network Cost						239 405 900 €

Tabela 1.60: Table with detailed description of CAPEX of Vasco's 2016 results.

High Traffic Scenario:

In this scenario we have to take into account the traffic calculated in **??**. In a first phase we will show the various existing topologies of the network. The first are the allowed topologies, physical and optical topologies, the second are the logical topology for all ODUs and finally the resulting physical topology.

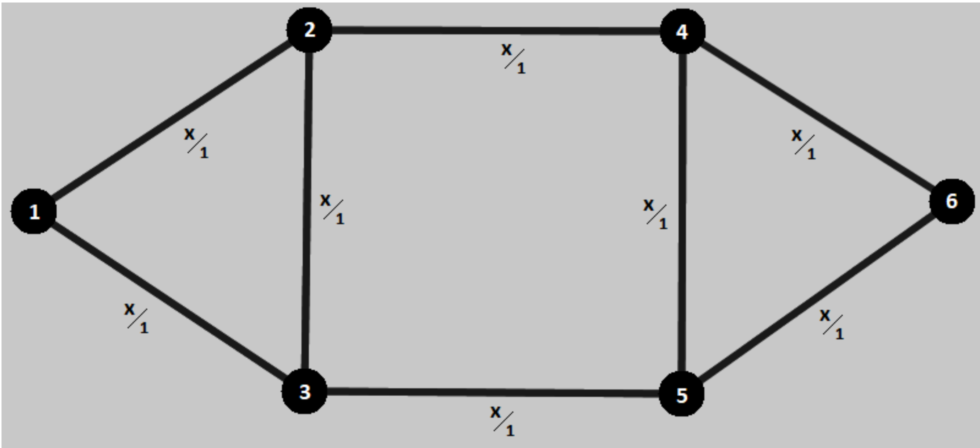

Figura 1.69: Allowed physical topology. The allowed physical topology is defined by the duct and sites in the field. It is assumed that each duct supports up to 1 bidirectional transmission system and each site supports up to 1 node.

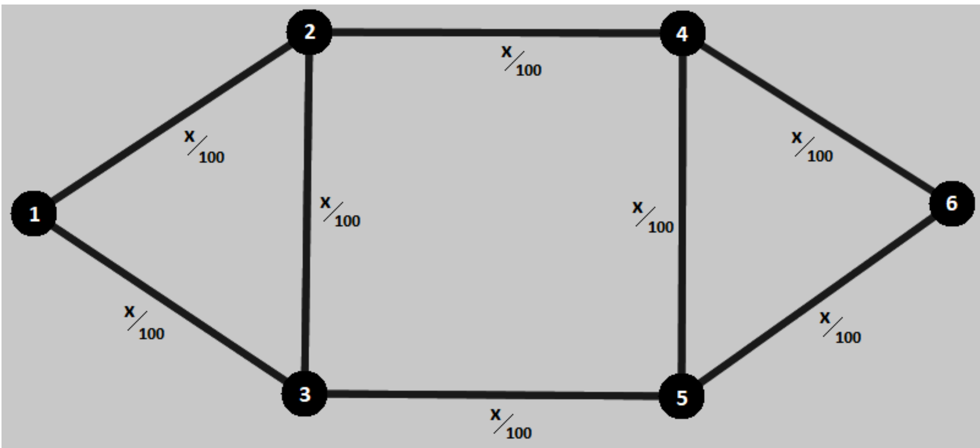

Figura 1.70: Allowed optical topology. The allowed optical topology is defined by the transport mode (opaque transport mode in this case). It is assumed that each transmission system supports up to 100 optical channels.

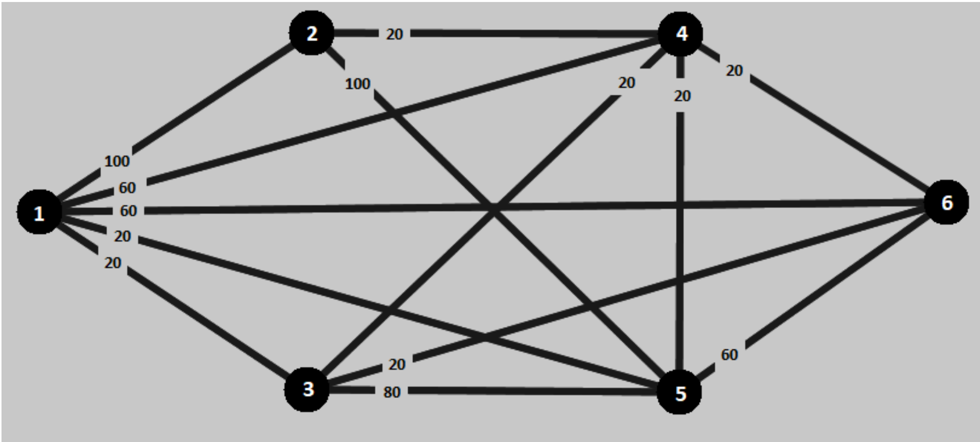

Figura 1.71: ODU0 logical topology defined by the ODU0 traffic matrix.

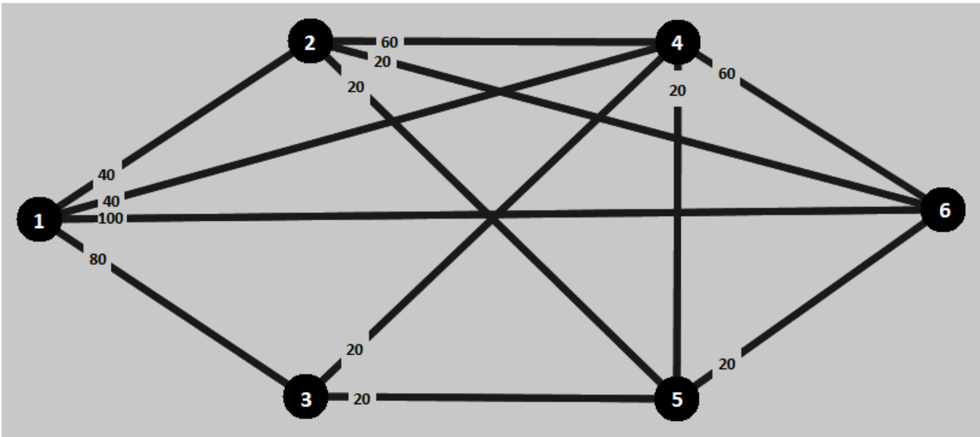

Figura 1.72: ODU1 logical topology defined by the ODU1 traffic matrix.

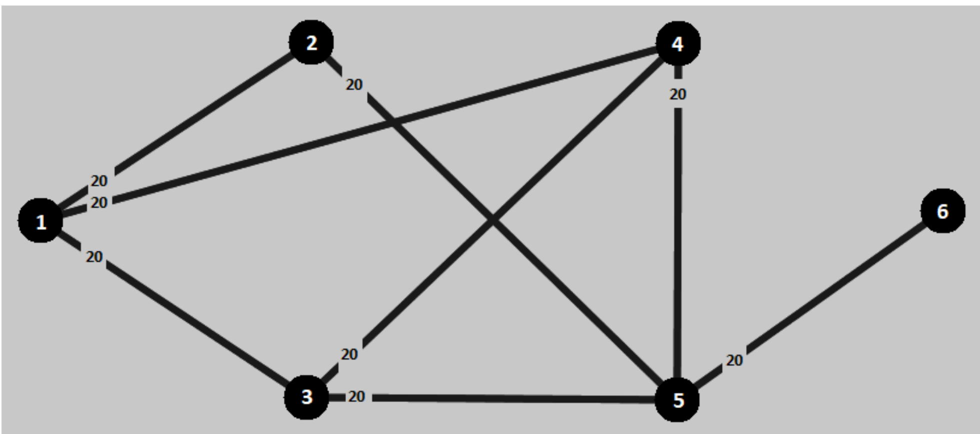

Figura 1.73: ODU2 logical topology defined by the ODU2 traffic matrix.

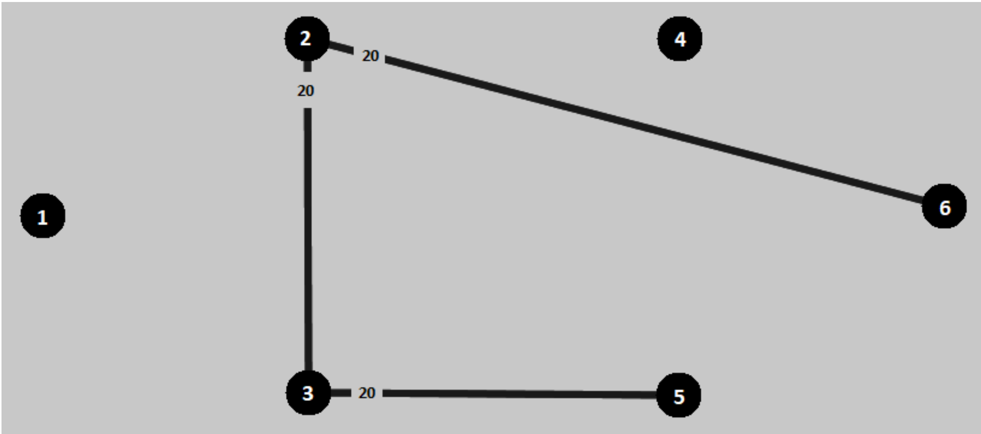

Figura 1.74: ODU3 logical topology defined by the ODU3 traffic matrix.

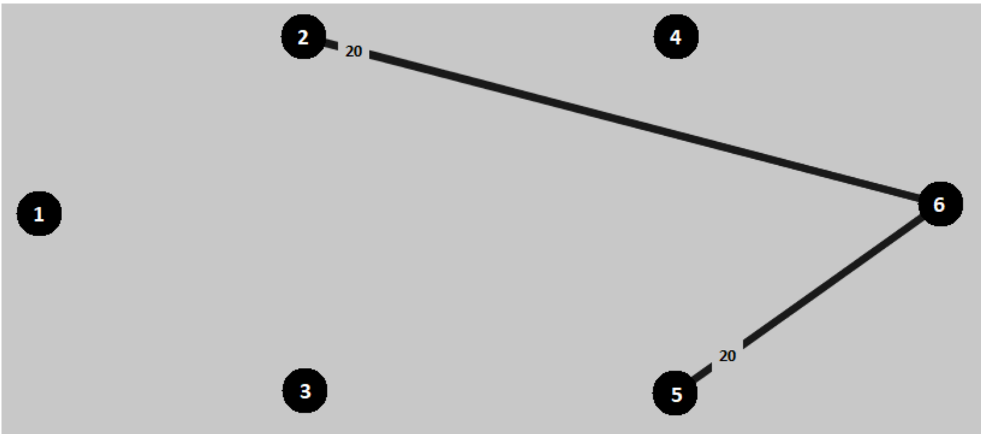

Figura 1.75: ODU4 logical topology defined by the ODU4 traffic matrix.

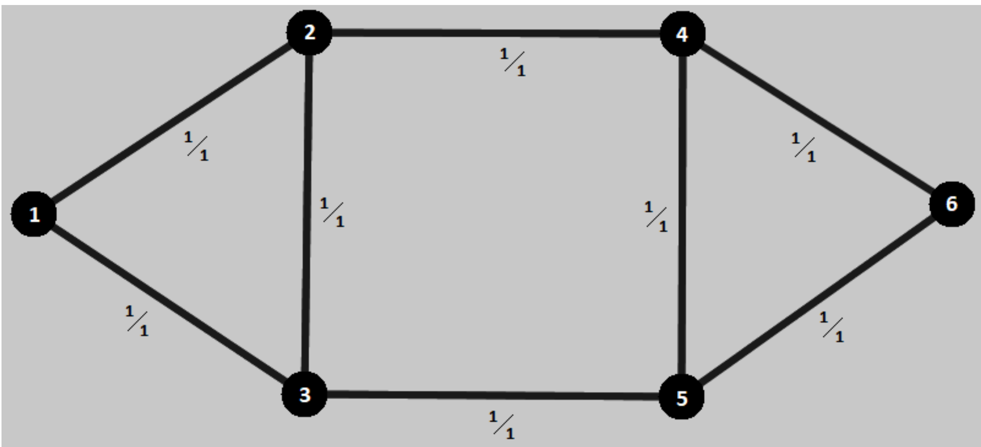

Figura 1.76: Physical topology after dimensioning.

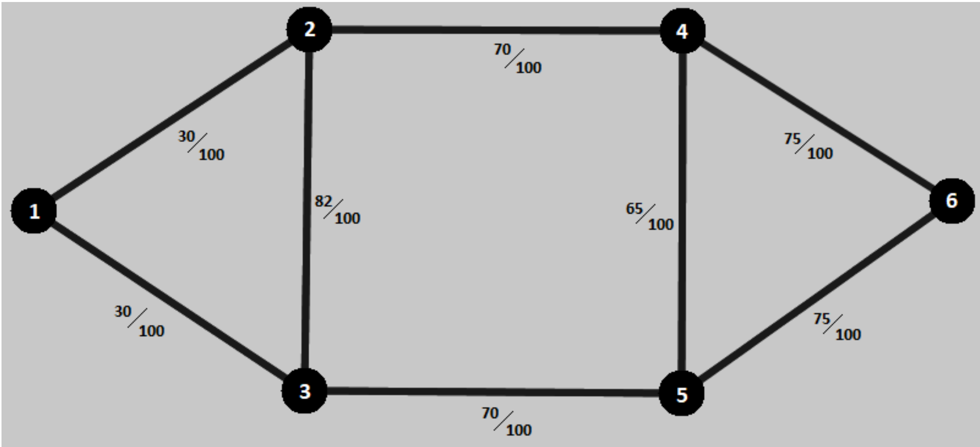

Figura 1.77: Optical topology after dimensioning.

Following all the steps mentioned in the **??**, applying the routing and grooming heuristic algorithms in the Net2Plan software and using all the data referring to this scenario, the obtained result for the Vasco's heuristics can be consulted in the following table 1.61. In table 1.5 mentioned in previous model we can see how all the values were calculated.

CAPEX of the Network						
			Quantity	Unit Price	Cost	Total
Link Cost	OLTs		16	15 000 €	240 000 €	397 520 000 €
	100 Gbits/s Transceivers		794	5 000 €/Gbit/s	397 000 000 €	
	Amplifiers		70	4 000 €	280 000 €	
Node Cost	Electrical	EXCs	6	10 000 €	60 000 €	79 511 800 €
		ODU0 Ports	1 200	10 €/port	12 000 €	
		ODU1 Ports	1 000	15 €/port	15 000 €	
		ODU2 Ports	320	30 €/port	9 600 €	
		ODU3 Ports	120	60 €/port	7 200 €	
		ODU4 Ports	80	100 €/port	8 000 €	
		Line Ports	794	100 000 €/port	79 400 000 €	
	Optical	OXCs	0	20 000 €	0 €	
		Ports	0	2 500 €/port	0 €	
Total Network Cost						477 031 800 €

Tabela 1.61: Table with detailed description of CAPEX of Vasco's 2016 results.

Conclusions

Once we have obtained the results for all the scenarios for opaque without survivability and opaque with 1+1 protection we will now draw some conclusions about these results. For a better analysis of the results will be created the table 1.62 with the number of line ports, tributary ports and transceivers because they are important values for the cost of CAPEX, the cost of links, the cost of nodes and finally the cost of CAPEX.

	Low Traffic	Medium Traffic	High Traffic
CAPEX without survivability	14 382 590 €	92 405 900 €	178 831 800 €
CAPEX/Gbit/s without survivability	28 765 €/Gbit/s	18 481 €/Gbit/s	17 883 €/Gbit/s
Traffic (Gbit/s)	500	5 000	10 000
Bidirectional Links used	8	8	8
Number of Line ports	46	398	794
Number of Tributary ports	136	1 360	2 720
Number of Transceivers	46	398	794
Link Cost	23 520 000 €	199 520 000 €	397 520 000 €
Node Cost	4 662 590 €	39 885 900 €	79 511 800 €
CAPEX	**28 182 590 €**	**239 405 900 €**	**477 031 800 €**
CAPEX/Gbit/s	**56 365 €/Gbit/s**	**47 881 €/Gbit/s**	**47 703 €/Gbit/s**

Tabela 1.62: Table with different value of CAPEX for this case.

Looking at the previous table we can make some comparisons between the opaque with 1+1 protection scenario:

- Comparing the low traffic with the others we can see that despite having an increase of factor ten (medium traffic) and factor twenty (high traffic), the same increase does not occur in the final cost (it is lower);

  This happens because the number of the transceivers is lower than expected which leads by carrying the traffic with less network components and, consequently, the network CAPEX is lower;

- Comparing the medium traffic with the high traffic we can see that the increase of the factor is double and in the final cost this factor is very close but still inferior;

  This happens because the number of the transceivers is also lower but very close to the expected;

- Comparing the CAPEX cost per bit we can see that in the low traffic the cost is higher than the medium and high traffic, which in these two cases the value is very similar;

  This happens because the lower the traffic, the higher CAPEX/bit will be. We can see that in medium and high traffic the results tend to be one closer value.

We can also make some comparisons between the opaque without survivability and opaque with 1+1 protection scenarios:

- We can see that in the opaque with 1+1 protection the CAPEX cost for all the three traffic is more than the double;

  This happens because in the opaque with 1+1 protection there is a need of having a primary and a backup path, in case of a network failure, and the backup path is typically longer and normally uses more than the double of the capacity of the primary;

- The number of the network components and the CAPEX cost are directly proportional to the traffic value. The higher the traffic value, the higher the network CAPEX cost;

  This happens because if the traffic value is higher, the network components have to be in more quantity to carry all the traffic end-to-end, both in the primary and backup paths;

- Comparing the CAPEX cost per bit we can see that has a similar case in both of the two scenarios. In the low traffic the cost is higher than the medium and high traffic, which in these two cases the value is very similar;

  This happens because the lower the traffic, the higher CAPEX/bit will be. We can see that in medium and high traffic the results tend to be one closer value.

Opens Issues

The creation of this model for any scenario, started with some considerations and some open issues being:

- Allow blocking.

  The presented model assume that the solution is possible or impossible, does not support a partial solution where some demands are not routed (are blocked);

- Allow multiple transmission system.

  The presented model for each link only supports one transmission system;

- Allowing multi-path routing.

  In the presented model all demands sharing the same end nodes have to follow the same path.

1.1.3 Transparent without Survivability

Student Name	:	Eduardo Fernandes (20/10/2018 -)
	:	Pedro Coelho (01/03/2018 -)
Goal	:	Implement the heuristic model for the transparent transport mode without survivability.

In the transparent transport mode (single-hop approach), the signals travel through the network in the optical domain between lightpaths. One advantage of this transport mode is that these networks require optical switching. This enables the realization of ongoing optical connections throughout several links without OEO (optical-electrical-optical) conversions. However, there are performed some conversions in some intermediate nodes.

Transparent optical connections creates lightpaths which require the assignment of a wavelength that will be used to be exchanged by wavelength converters in order to optimize the network and minimize the total CAPEX.

After the creation of the matrices and the network topology, it is necessary to apply the routing and grooming algorithms created. In the end, a report algorithm will be applied to obtain the best CAPEX result for the network in question.

We also must take into account the following particularity of this mode of transport:

- $N_{OXC,n}$ = 1, $\forall$ n that process traffic
- $N_{EXC,n}$ = 1, $\forall$ n that process traffic

The minimization of the network CAPEX is made through the equation 1.1 where in this case for the cost of nodes we have in consideration the electric cost 1.4 and the optical cost 1.5.

In this case the value of $P_{exc,c,n}$ is obtained by equation 1.10 for short-reach and by the equation 1.11 for long-reach and the value of $P_{oxc,n}$ is obtained by equation 1.12.

The equation 1.10 refers to the number of short-reach ports of the electrical switch with bit-rate c in node n, $P_{exc,c,n}$, i.e. the number of tributary ports with bit-rate c in node n which can be calculated as

$$P_{exc,c,n} = \sum_{d=1}^{N} D_{nd,c} \tag{1.10}$$

where $D_{nd,c}$ are the client demands between nodes n and d with bit rate c.

In this case there is the following particularity:

- When n=d the value of client demands is always zero, i.e, $D_{nn,c} = 0$

As previously mentioned, the equation 1.11 refers to the number of long-reach ports of the electrical switch with bit-rate -1 in node n, $P_{exc,-1,n}$, i.e. the number of add ports of node n which can be calculated as

$$P_{exc,-1,n} = \sum_{j=1}^{N} f_{nj}^{od} \tag{1.11}$$

where f_{nj}^{od} is the number of optical channels between node n and node j for all demand pairs (od).

The equation 1.12 refers to the number of line ports and the number of adding ports of node n which can be calculated as

$$P_{oxc,n} = \sum_{j=1}^{N} 2f_{nj}^{od} + \sum_{j=1}^{N} \lambda_{nj} \tag{1.12}$$

where f_{nj}^{od} refers to the number of line ports for all demand pairs (od) and λ_{nj} refers to the number of add ports.

To implement this heuristic approach there are used algorithms made in Java in a programming software called Eclipse and they are tested in an open-source network program called Net2Plan. In the Net2Plan guide section **??** there is an explanation on how to use and test them in this network planner.

In the next pages it will be described all the steps performed to obtain the final results in the transparent transport mode without survivability. In the figure below 1.78 it is shown a fluxogram with the description of this transport mode approach.

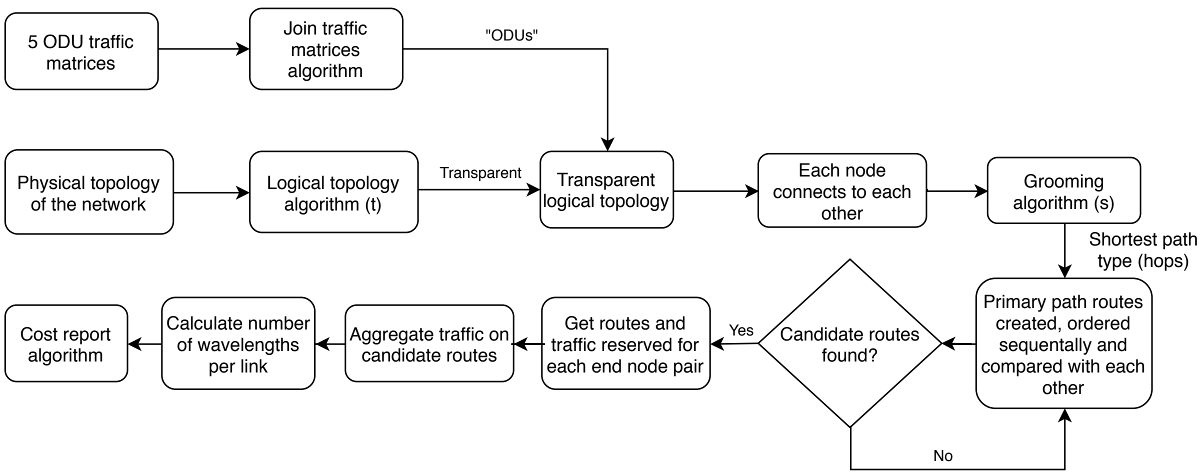

Figura 1.78: Fluxogram with the steps performed in the transparent without survivability transport mode approach.

Creation and join the traffic matrices

The first step is to create the traffic matrices based on the reference network **??**. In order to create the 5 traffic matrices in Net2Plan it is necessary the length of all the links and the total traffic used in this network, so later it is needed to define in Net2Plan the length in all end nodes and the total traffic depends on the value of traffic used (low traffic - 0.5 Tbit/s, medium traffic - 5 Tbit/s and high traffic - 10 Tbit/s). As you can see in the figure below, it is defined the path of the 5 ODUs and they will be aggregated in just one single ODU, making it possible to join all the demands in just one file and load it later into the network. This final resulting ODU joins the multiple traffic demands from all the traffic matrices previously created and, of course, the traffic demands will depend on the values used on the creation of the matrices (low, medium and high traffic).

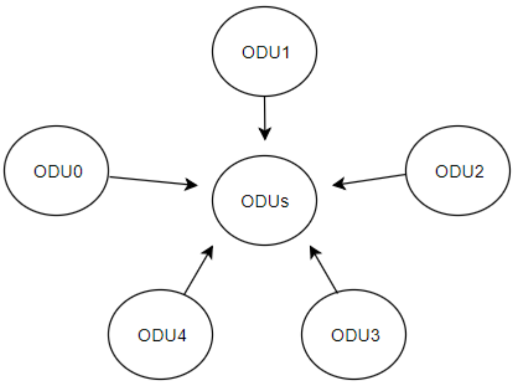

Figura 1.79: Join the 5 ODU traffic matrices into 1 single file "ODUs". The 5 traffic demands from the traffic matrices previously created are joined into 1 file to load it later on Net2Plan.

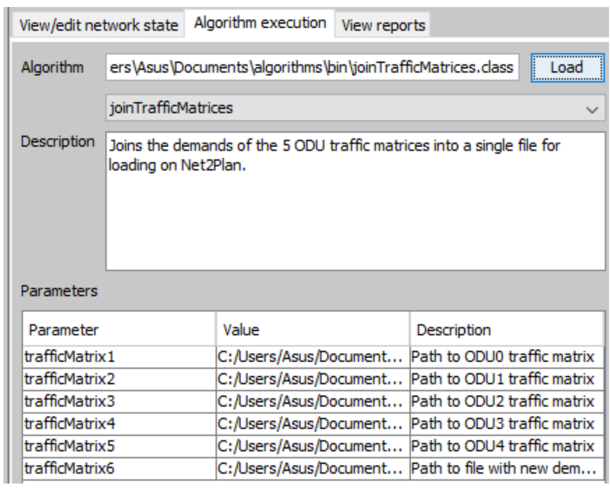

Figura 1.80: Load of the join traffic matrices algorithm for the transparent transport mode on Net2Plan. It is defined the 5 paths to load the 5 ODU traffic matrices and the last path is the one where will be saved the file that joins all 5 the traffic demands.

Creation of the physical topology

The next step is to create the allowed physical topology of the network in Net2Plan. This network consists in 6 nodes and 8 bidirectional links. It is now also possible to define the length in all links. In the figure below it is shown the allowed physical topology in this transport mode.

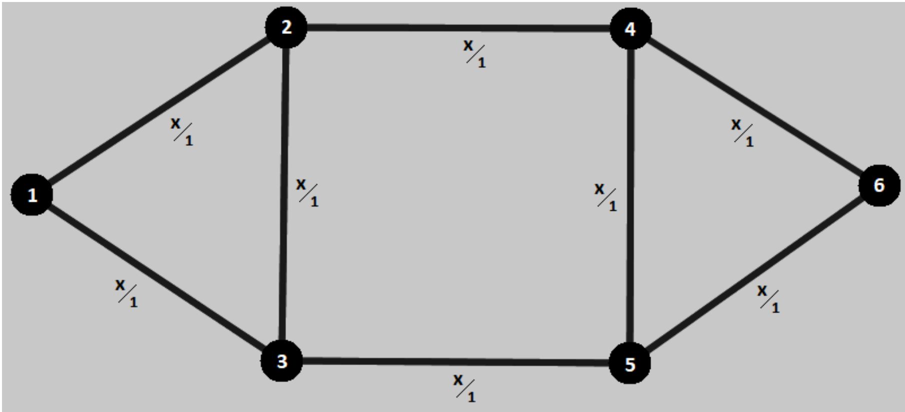

Figura 1.81: Allowed physical topology. The allowed physical topology is defined by the duct and sites in the field. It is assumed that each duct supports up to 1 bidirectional transmission system and each site supports up to 1 node.

Creation of the logical topology

It is now time to create the allowed logical topology. A network topology represents how the links and the nodes of the network interconnect with each other and the logical topology algorithm creates the logical topology on another layer. In the transparent transport mode each node connects to each other creating direct links between all nodes in the network. Going through all nodes, if a node has a different index from other one, then creates a shortest and direct link between them. These additions of links between end nodes are made in the new upper layer of the network. The respective demands are saved in the new upper layer and those demands from the lower layer are then removed. The lower layer is the physical layer of the network and it is now created a new upper layer which is the logical layer of the network and represents the logical topology of the transparent transport mode. The allowed physical and optical topologies, the logical topologies for all ODUs and the resulting physical topology is shown in the next section below 1.1.3 for the three traffic scenarios. It is shown below three figures with the code in Java of the creation of the network logical topology, the load of the logical topology algorithm in Net2Plan and the resulting allowed optical topology for the transparent transport mode without survivability.

```java
if(netPlan.isSingleLayer() && logicalTopology.equalsIgnoreCase("Transparent"))
{
    this.lowerLayer = netPlan.getNetworkLayerDefault();
    lowerLayer.setName("Physical Topology");
    this.upperLayer = netPlan.addLayer("Logical Topology Transparent","Upper layer of the design","ODU","ODU",null);
    upperLayer.setDescription("Transparent Logical Topology");
    netPlan.removeAllLinks(upperLayer);

    for (Node i : netPlan.getNodes()) {
        for (Node j : netPlan.getNodes()) {
            if (i.getIndex() != j.getIndex())
            {
                netPlan.addLink(i, j, 0 , netPlan.getNodePairEuclideanDistance(i, j), 200000 , null , upperLayer);
            }
        }
    }
}
```

Figura 1.82: Java code of the logical topology approach for the transparent transport mode. The logical layer is created by adding direct links between all end nodes. The new layer is now the transparent logical topology of the network.

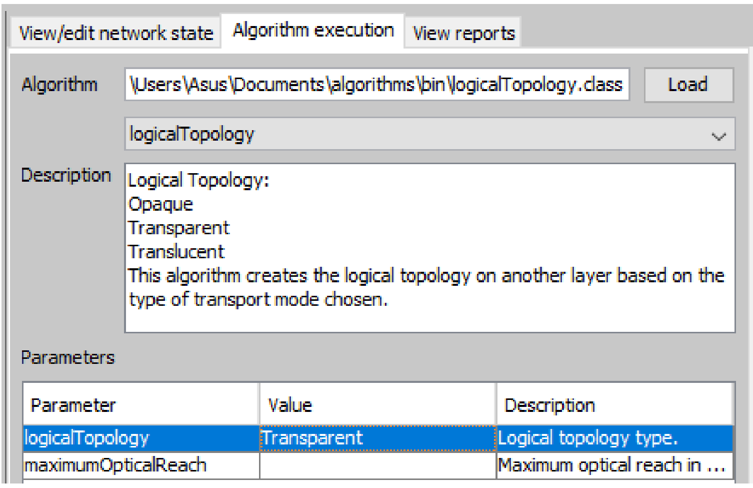

Figura 1.83: Load of the logical topology algorithm for the transparent transport mode.

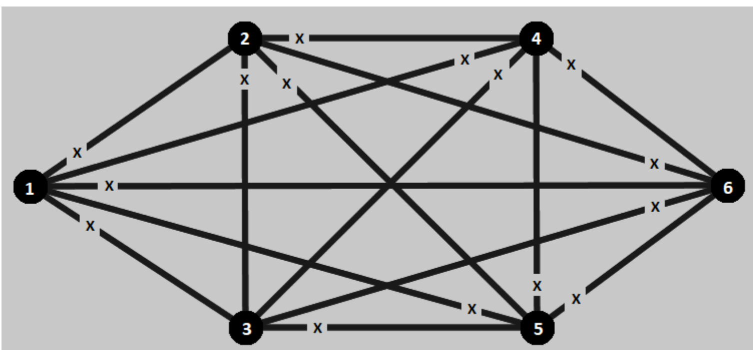

Figura 1.84: Allowed optical topology. It is assumed that each connections between demands supports up to 100 lightpaths.

Creation of routes and aggregation of traffic

After a network topology is created, it is now time to set the routing algorithm. In the transparent without survivability transport mode the routing algorithm is similar with the one used in opaque transport mode. It starts with going through all the demands and nodes which have different index between them (end nodes), create bidirectional routes (in this case the primary paths) based on the shortest path Dijkstra algorithm and then search the candidate routes for the respective demand. In this report it is used the shortest path type in hops. These routes are ordered sequentially and the shortest one per each demand is the primary path. The demands from the lower layer are removed and then saved in the upper layer. After this step, the routes are saved to a "Set" of routes and in each link of end nodes it is set the traffic demands into these routes that will integrate the whole network.

```
case "Logical Topology Transparent":

    for (Demand d : netPlan.getDemands(lowerLayer)) {
        boolean odd=true;
        int counter=0;

        Set<Route> droutes = d.getRoutes();
        System.out.println("droutes: " + droutes.size());

        for(Route c: droutes) {
            counter++;
            boolean jump=false;

            if(odd) {
                c.setCarriedTraffic(d.getOfferedTraffic(), d.getOfferedTraffic());
                save=c;
                odd=false;
                System.out.println("Roots");
            }
```

Figura 1.85: Creation of routes and aggregation of traffic for the transparent without survivability transport mode. The candidate routes are searched by the shortest path type method and the offered traffic demands are set into these routes.

```
ArrayList<Long> tNodeIds = netPlan.getNodeIds();
Node in;
Node out;
Set<Route> groomRoute;
Set<ProtectionSegment> protectRoutes;
Route compare=null;
ProtectionSegment compare1=null;
List<Link> path;
int nW=0;

for (long tNodeId : tNodeIds)   {
    in = netPlan.getNodeFromId(tNodeId);
    for (long tNodeId1 : tNodeIds)  {

        if(tNodeId==tNodeId1)continue;

        out = netPlan.getNodeFromId(tNodeId1);
        double totaltraffic=0;

        groomRoute=netPlan.getNodePairRoutes(in,out,false,lowerLayer);
        protectRoutes=netPlan.getNodePairProtectionSegments(in,out,false,lowerLayer);

        for(Route d:groomRoute)
        {
            totaltraffic = totaltraffic + d.getCarriedTraffic();
            compare=d;
        }
```

Figura 1.86: Creation of routes and aggregation of traffic for the transparent without survivability transport mode. The traffic demands are set into the candidate primary path routes found earlier.

Function	Definition
netPlan.getDemands(lowerLayer)	Returns the array of demands for the lower layer.
d.getRoutes()	Returns all the routes associated to the demand "d".
c.setCarriedTraffic()	Sets the route carried traffic and the occupied capacity in the links, setting it up to be the same in all links.
d.getOfferedTraffic()	Returns the offered traffic of the demand "d".
netPlan.getNodeIds()	Returns the array of the nodes' indexes.
netPlan.getNodeFromId(tNodeId)	Returns the node with the index "tNodeId".
netPlan.getNodePairRoutes (in,out,false,lowerLayer)	Returns the routes at "lowerLayer" from nodes "in" and "out".

Tabela 1.63: Table with the description of the main functions in the creation of routes and aggregation of traffic in the grooming algorithm.

Calculation of the number of wavelengths per link

The final step of the routing and grooming algorithms is to calculate the number of wavelengths per link for the whole network. This is the last and an important step because with the number of wavelengths per link in the network, it is possible to calculate other network components. In the transparent transport mode, as in the figure below shows, the algorithm starts with going through all the nodes which have different index between them (end nodes) and in all the links that crosses between these pairs of nodes is reserved a link capacity based on the previous traffic aggregation. The total carried traffic in the link including protection and non-protection segments will be divided by the wavelength capacity and it is now possible to obtain the number of wavelengths per link.

```
for (Link link:path)
{
    String nw = link.getAttribute("nW");
    nW=0;

    if(nw!=null)
    {
        nW = Integer.parseInt(nw);
        nW = (int) (nW+Math.ceil(totaltraffic/wavelengthCapacity));
        link.setAttribute("nW",String.valueOf(nW));
    }else {
        nW = (int) Math.ceil(totaltraffic/wavelengthCapacity);
        link.setAttribute("nW",String.valueOf(nW));
    }
}
```

Figura 1.87: Calculation of the number of wavelengths per link for the transparent transport mode. The link capacity is reserved based on the previous traffic aggregation.

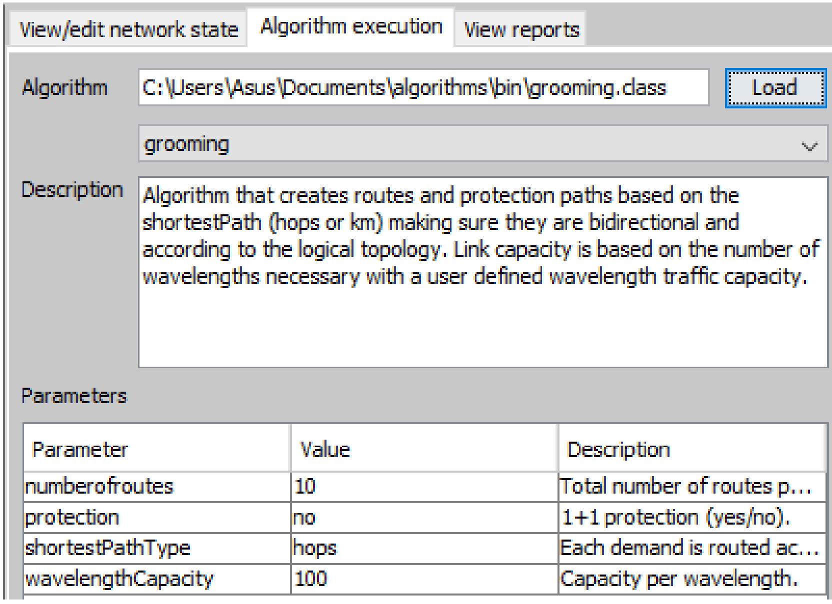

Figura 1.88: Load of the grooming algorithm for the transparent without survivability transport mode. The total number of routes per demand is set to 10, the user can define if the model is with or without protection, the shortest path type is set to "hops" and the capacity per wavelength is used 100 optical channels.

Network cost report

In order to obtain the network CAPEX results, the formulas needed to calculate the network elements and that are demonstrated previously in the beginning of this section 1.1.3 were "translated" into Java code in a cost report algorithm. This algorithm can be loaded in Net2Plan and calculates and shows in tables the network CAPEX and also the per-link and per-node information with more details.

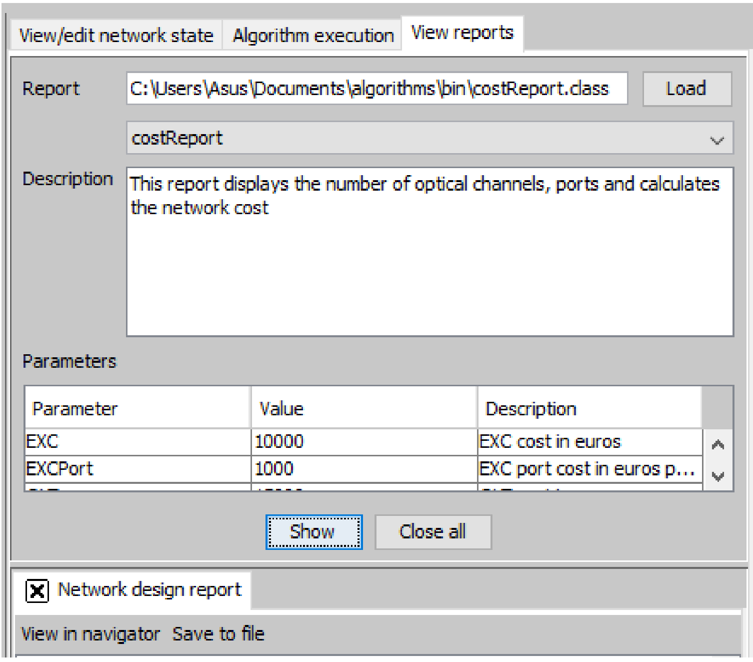

Figura 1.89: Load of the cost report algorithm on Net2Plan. The result view is an HTML page with the network optical and electrical components and their costs.

Result description

It is already known all the necessary formulas to obtain the CAPEX value for the reference network **??**. As described in the subsection of the network traffic **??**, it is necessary to obtain three different values of CAPEX for the low (0.5 Tbit/s), medium (5 Tbit/s) and high (10 Tbit/s) traffic. It is used a network software program called Net2Plan which can design the traffic matrices, create all the network topologies, simulate the algorithms into the network implemented in the programming software called Eclipse and analyze the results obtained. In this chapter will be demonstrated the results by Vasco's heuristics from 2016. In each of the three traffic scenarios, it will be shown the network topologies followed by the table with the CAPEX value of the network.

Low Traffic Scenario:

In this scenario we have to take into account the traffic calculated in **??**. In a first phase we will show the various existing topologies of the network. The first are the allowed topologies, physical and optical topologies, the second are the logical topology for all ODUs and finally the resulting physical topology.

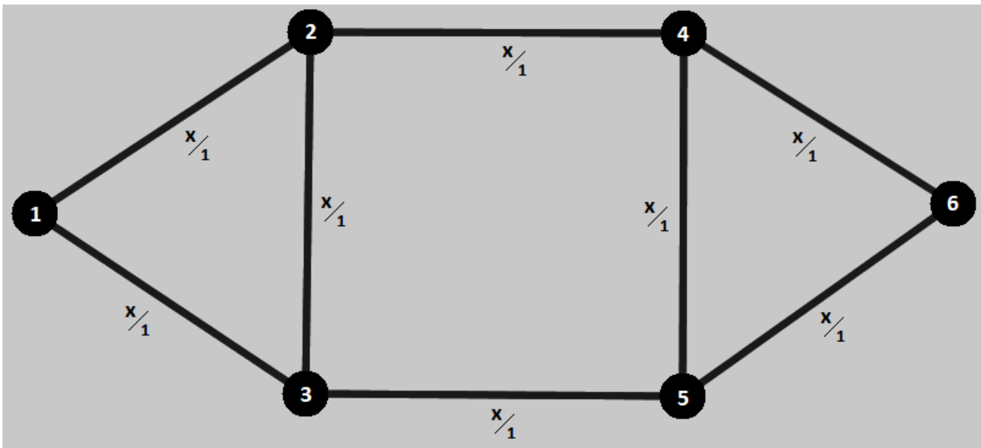

Figura 1.90: Allowed physical topology. The allowed physical topology is defined by the duct and sites in the field. It is assumed that each duct supports up to 1 bidirectional transmission system and each site supports up to 1 node.

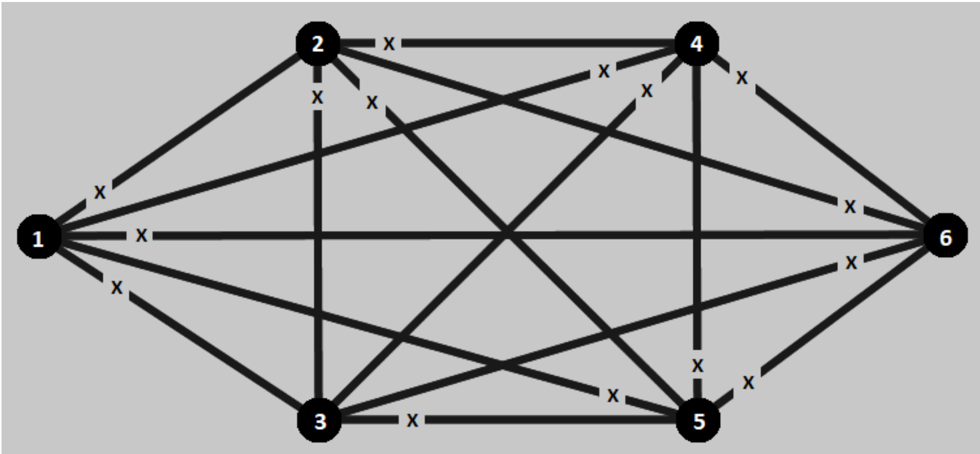

Figura 1.91: Allowed optical topology. The allowed optical topology is defined by the transport mode (transparent transport mode in this case). It is assumed that each connections between demands supports up to 100 lightpaths.

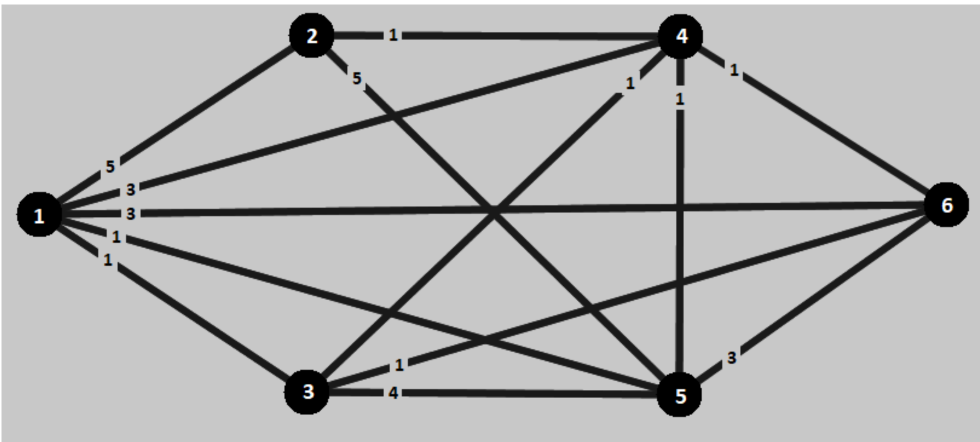

Figura 1.92: ODU0 logical topology defined by the ODU0 traffic matrix.

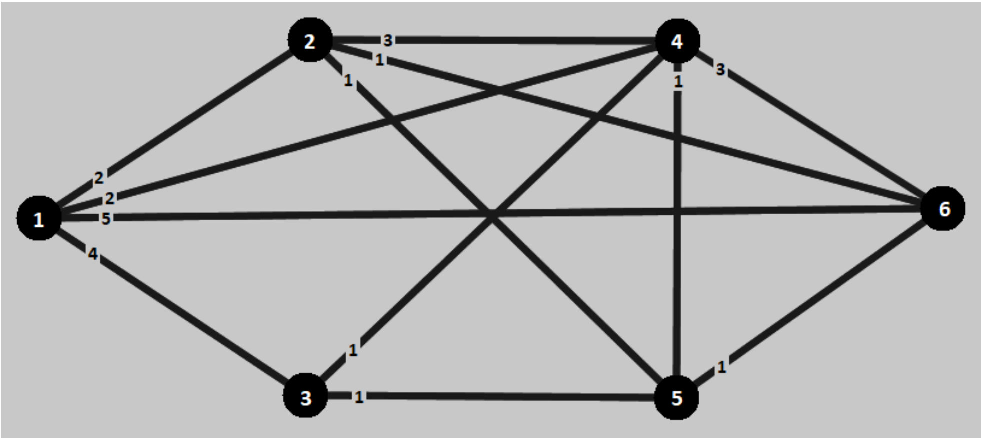

Figura 1.93: ODU1 logical topology defined by the ODU1 traffic matrix.

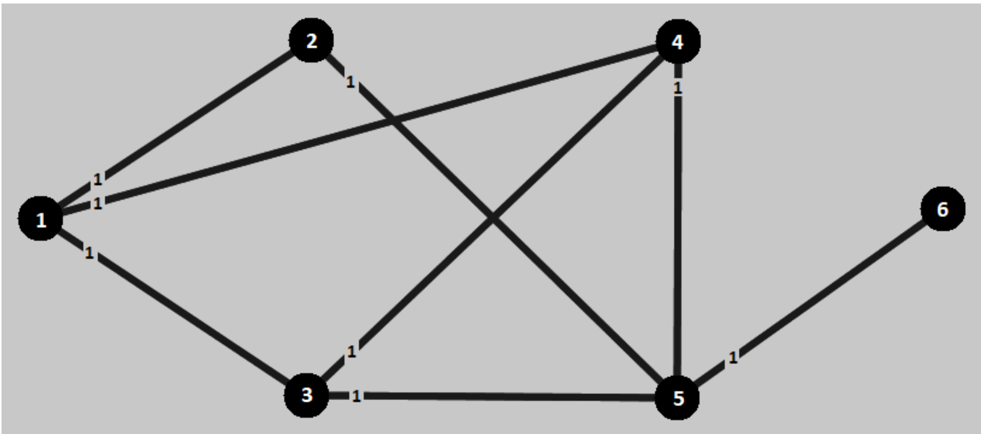

Figura 1.94: ODU2 logical topology defined by the ODU2 traffic matrix.

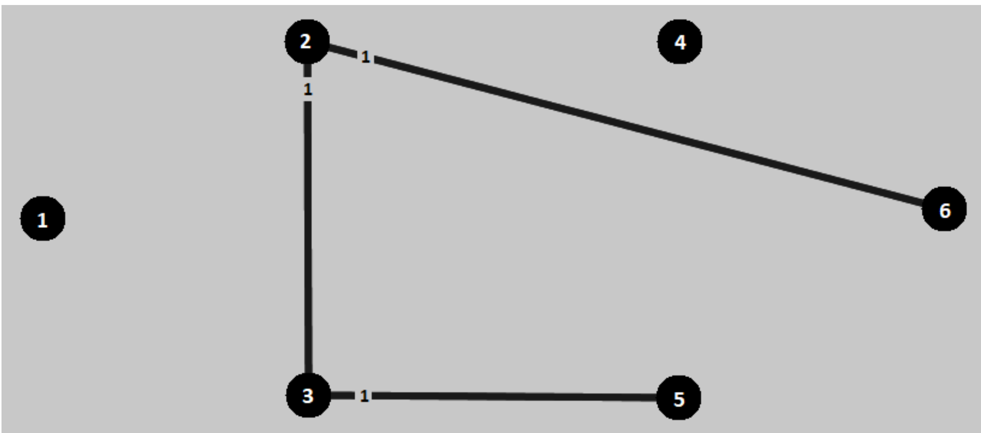

Figura 1.95: ODU3 logical topology defined by the ODU3 traffic matrix.

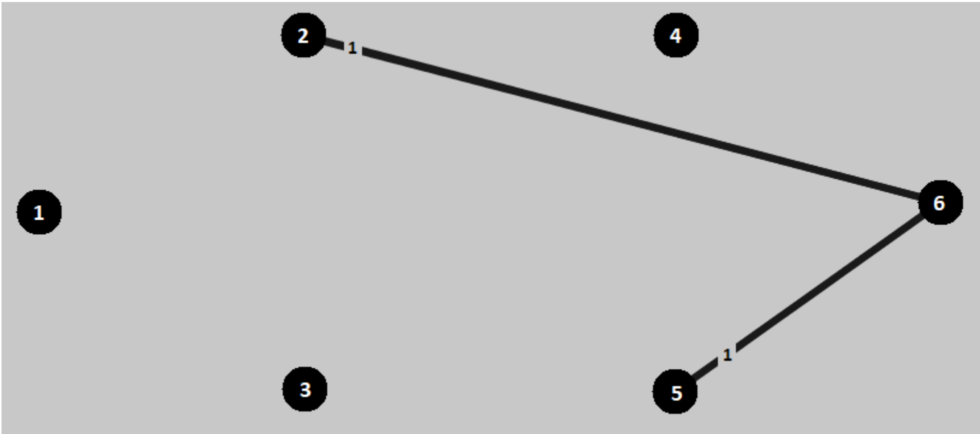

Figura 1.96: ODU4 logical topology defined by the ODU4 traffic matrix.

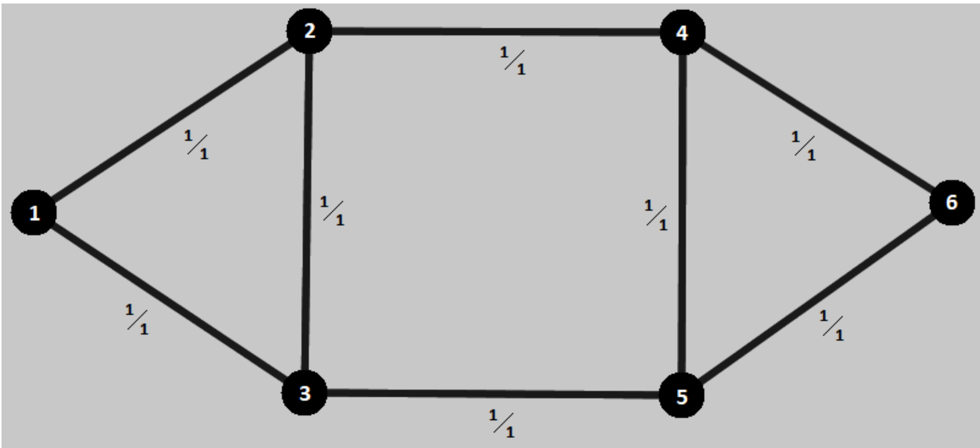

Figura 1.97: Physical topology after dimensioning.

Following all the steps mentioned in the **??**, applying the routing and grooming heuristic algorithms in the Net2Plan software and using all the data referring to this scenario, the obtained result for the Vasco's heuristics can be consulted in the following table 1.64.

CAPEX of the Network						
			Quantity	Unit Price	Cost	Total
Link Cost	OLTs		16	15 000 €	240 000 €	26 520 000 €
	100 Gbits/s Transceivers		52	5 000 €/Gbit/s	26 000 000 €	
	Amplifiers		70	4 000 €	280 000 €	
Node Cost	Electrical	EXCs	6	10 000 €	60 000 €	3 797 590 €
		ODU0 Ports	60	10 €/port	600 €	
		ODU1 Ports	50	15 €/port	750 €	
		ODU2 Ports	16	30 €/port	480 €	
		ODU3 Ports	6	60 €/port	360 €	
		ODU4 Ports	4	100 €/port	400 €	
		Transponders	34	100 000 €/port	3 400 000 €	
	Optical	OXCs	6	20 000 €	120 000 €	
		Line Ports	52	2 500 €/port	130 000 €	
		Add Ports	34	2 500 €/port	85 000 €	
Total Network Cost						30 317 590 €

Tabela 1.64: Table with detailed description of CAPEX of Vasco's 2016 results.

All the values calculated in the previous table were obtained through the equations 1.2 and 1.3 referred to in section 1.1, but for a more detailed analysis we created table 1.65 where we can see how all the parameters are calculated individually.

Medium Traffic Scenario:

In this scenario we have to take into account the traffic calculated in **??**. In a first phase we will show the various existing topologies of the network. The first are the allowed topologies, physical and optical topologies, the second are the logical topology for all ODUs and finally the resulting physical topology.

	Equation used to calculate the cost
OLTs	$2\sum_{i=1}^{N}\sum_{j=i+1}^{N} L_{ij}\gamma_0^{OLT}$
Transceivers	$2\sum_{i=1}^{N}\sum_{j=i+1}^{N} L_{ij}f_{ij}^{od}\gamma_1^{OLT}\tau$
Amplifiers	$2\sum_{i=1}^{N}\sum_{j=i+1}^{N} L_{ij}N_{ij}^{R}c^{R}$
EXCs	$\sum_{n=1}^{N} N_{exc,n}\gamma_{e0}$
ODU0 Port	$\sum_{n=1}^{N}\sum_{d=1}^{N} N_{exc,n}D_{nd,0}\gamma_{e1,0}$
ODU1 Port	$\sum_{n=1}^{N}\sum_{d=1}^{N} N_{exc,n}D_{nd,1}\gamma_{e1,1}$
ODU2 Port	$\sum_{n=1}^{N}\sum_{d=1}^{N} N_{exc,n}D_{nd,2}\gamma_{e1,2}$
ODU3 Port	$\sum_{n=1}^{N}\sum_{d=1}^{N} N_{exc,n}D_{nd,3}\gamma_{e1,3}$
ODU4 Port	$\sum_{n=1}^{N}\sum_{d=1}^{N} N_{exc,n}D_{nd,4}\gamma_{e1,4}$
LR Transponders	$\sum_{n=1}^{N}\sum_{j=1}^{N} N_{exc,n}\lambda_{od}\gamma_{e1,-1}$
OXCs	$\sum_{n=1}^{N} N_{oxc,n}\gamma_{o0}$
Add Port	$\sum_{n=1}^{N}\sum_{j=1}^{N} N_{oxc,n}\lambda_{od}\gamma_{o1}$
Line Port	$\sum_{n=1}^{N}\sum_{j=1}^{N} N_{oxc,n}f_{ij}^{od}\gamma_{o1}$
CAPEX	The final cost is calculated by summing all previous results.

Tabela 1.65: Table with description of calculation

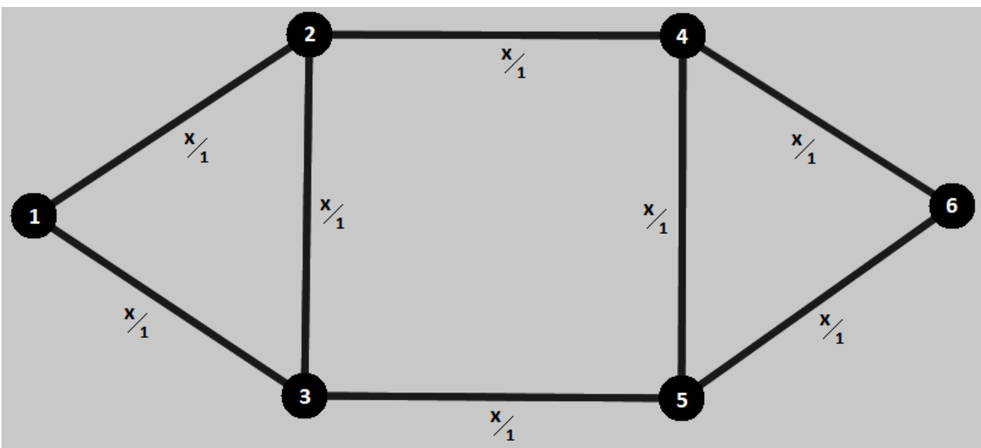

Figura 1.98: Allowed physical topology. The allowed physical topology is defined by the duct and sites in the field. It is assumed that each duct supports up to 1 bidirectional

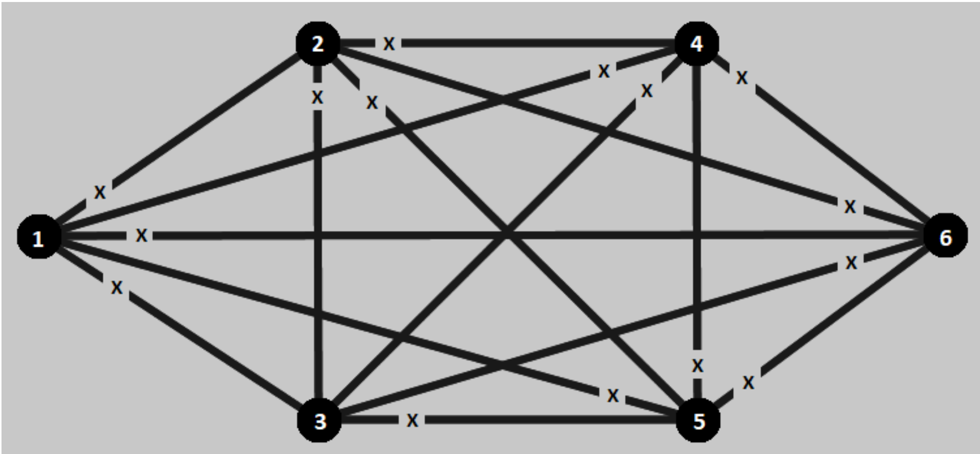

Figura 1.99: Allowed optical topology. The allowed optical topology is defined by the transport mode (transparent transport mode in this case). It is assumed that each connections between demands supports up to 100 lightpaths.

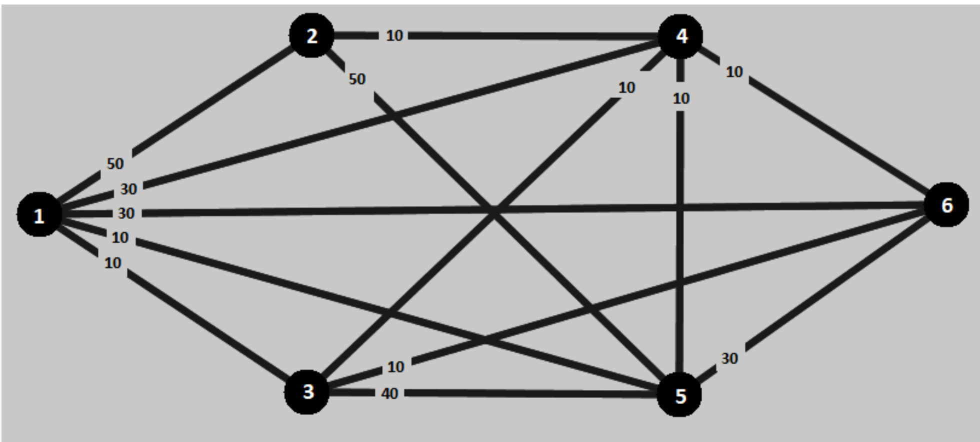

Figura 1.100: ODU0 logical topology defined by the ODU0 traffic matrix.

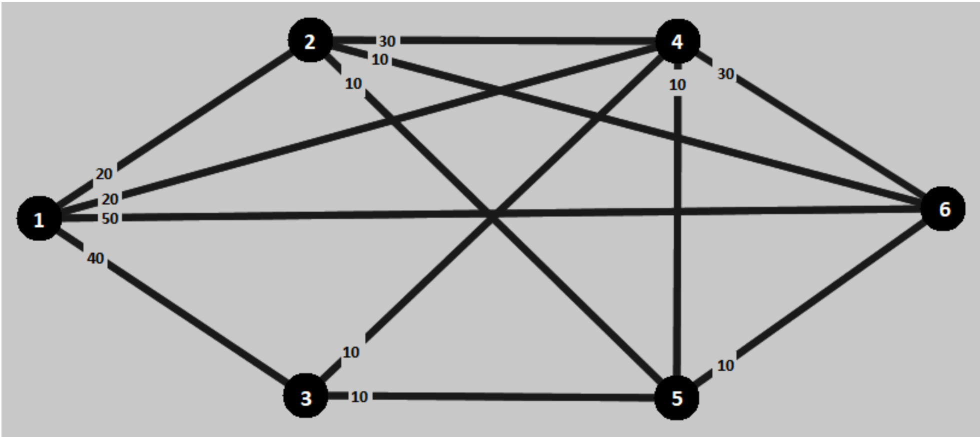

Figura 1.101: ODU1 logical topology defined by the ODU1 traffic matrix.

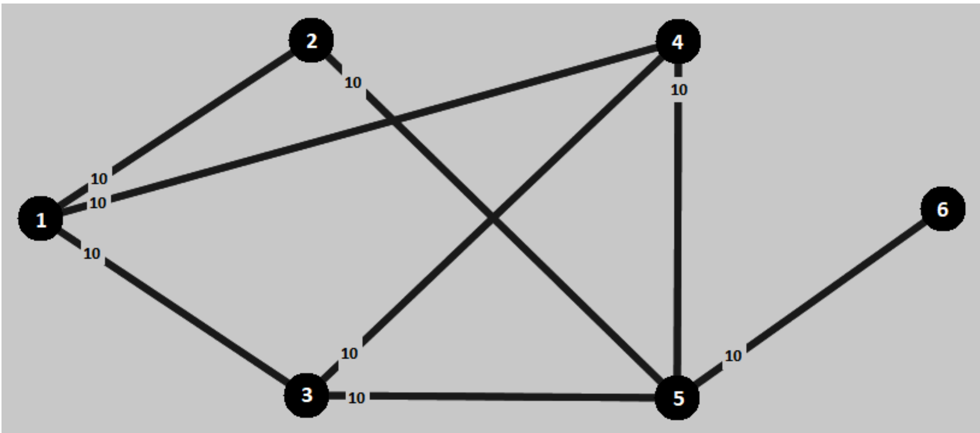

Figura 1.102: ODU2 logical topology defined by the ODU2 traffic matrix.

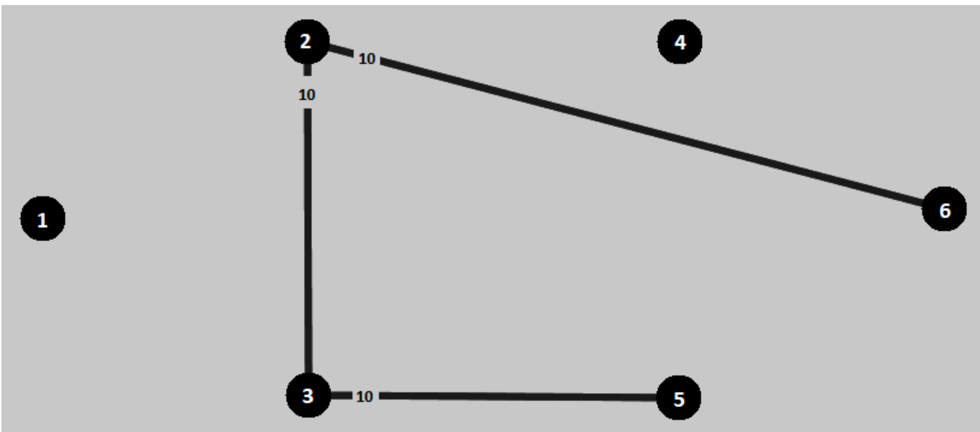

Figura 1.103: ODU3 logical topology defined by the ODU3 traffic matrix.

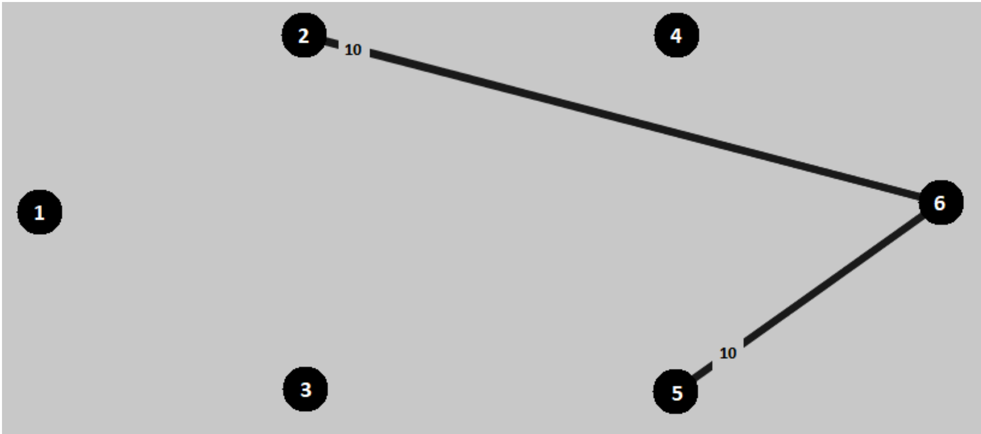

Figura 1.104: ODU4 logical topology defined by the ODU4 traffic matrix.

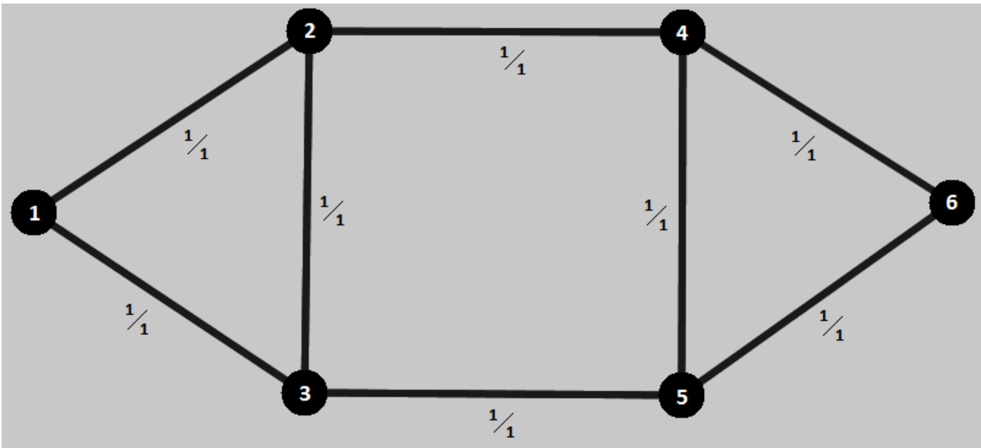

Figura 1.105: Physical topology after dimensioning.

Following all the steps mentioned in the **??**, applying the routing and grooming heuristic algorithms in the Net2Plan software and using all the data referring to this scenario, the obtained result for the Vasco's heuristics can be consulted in the following table 1.66. In table 1.65 mentioned in previous scenario we can see how all the values were calculated.

CAPEX of the Network						
			Quantity	Unit Price	Cost	Total
Link Cost	OLTs		16	15 000 €	240 000 €	84 520 000 €
	100 Gbits/s Transceivers		168	5 000 €/Gbit/s	84 000 000 €	
	Amplifiers		70	4 000 €	280 000 €	
Node Cost	Electrical	EXCs	6	10 000 €	60 000 €	15 180 900 €
		ODU0 Ports	600	10 €/port	6 000 €	
		ODU1 Ports	500	15 €/port	7 500 €	
		ODU2 Ports	160	30 €/port	4 800 €	
		ODU3 Ports	60	60 €/port	3 600 €	
		ODU4 Ports	40	100 €/port	4 000 €	
		Transponders	142	100 000 €/port	14 200 000 €	
	Optical	OXCs	6	20 000 €	120 000 €	
		Line Ports	168	2 500 €/port	420 000 €	
		Add Ports	142	2 500 €/port	355 000 €	
Total Network Cost						99 700 900 €

Tabela 1.66: Table with detailed description of CAPEX of Vasco's 2016 results.

High Traffic Scenario:

In this scenario we have to take into account the traffic calculated in **??**. In a first phase we will show the various existing topologies of the network. The first are the allowed topologies, physical and optical topologies, the second are the logical topology for all ODUs and finally the resulting physical topology.

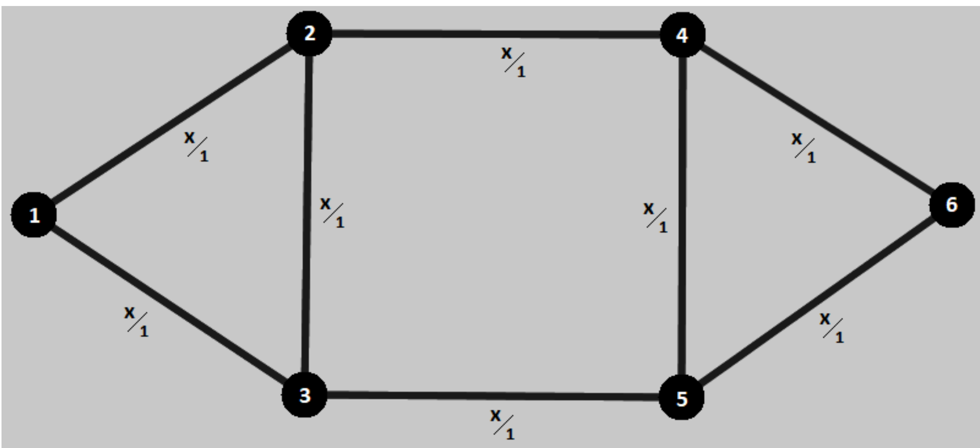

Figura 1.106: Allowed physical topology. The allowed physical topology is defined by the duct and sites in the field. It is assumed that each duct supports up to 1 bidirectional transmission system and each site supports up to 1 node.

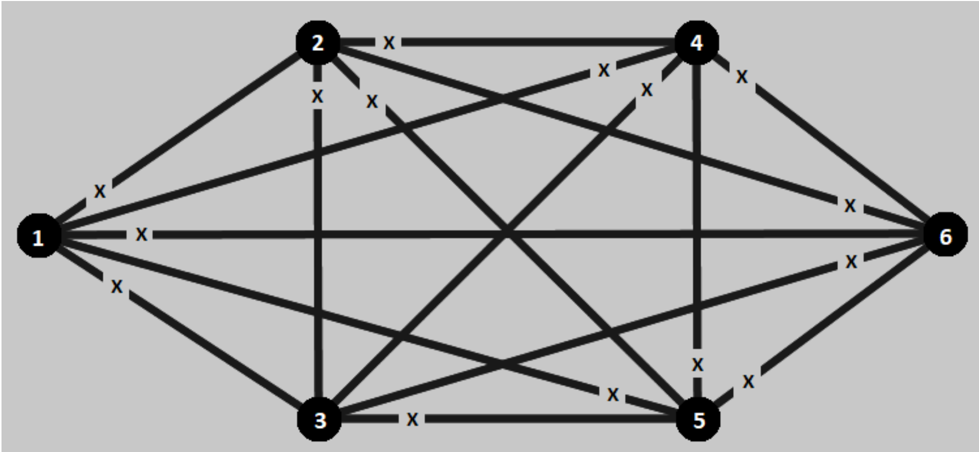

Figura 1.107: Allowed optical topology. The allowed optical topology is defined by the transport mode (transparent transport mode in this case). It is assumed that each connections between demands supports up to 100 lightpaths.

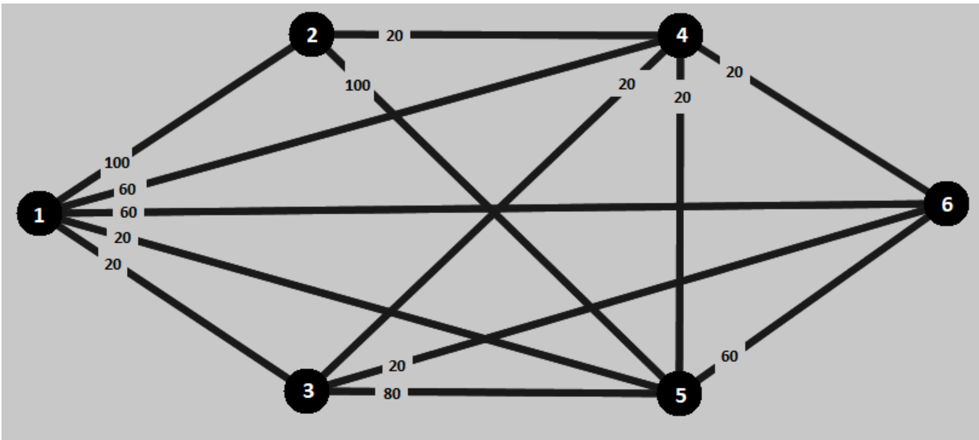

Figura 1.108: ODU0 logical topology defined by the ODU0 traffic matrix.

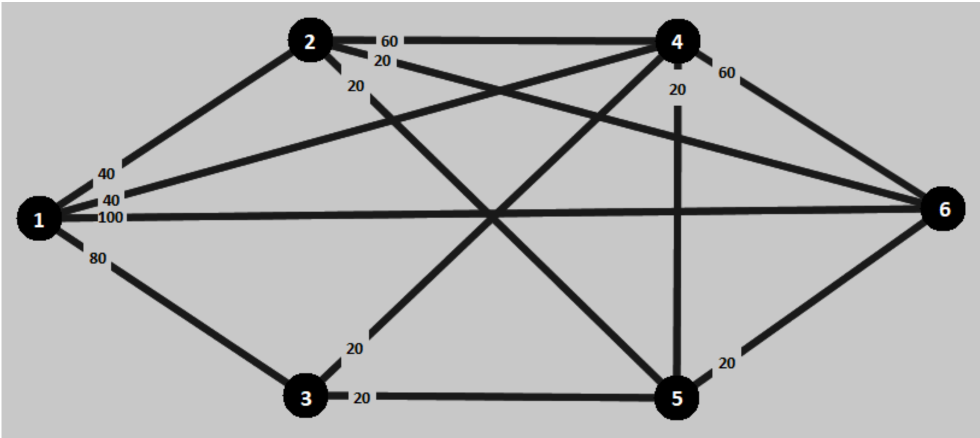

Figura 1.109: ODU1 logical topology defined by the ODU1 traffic matrix.

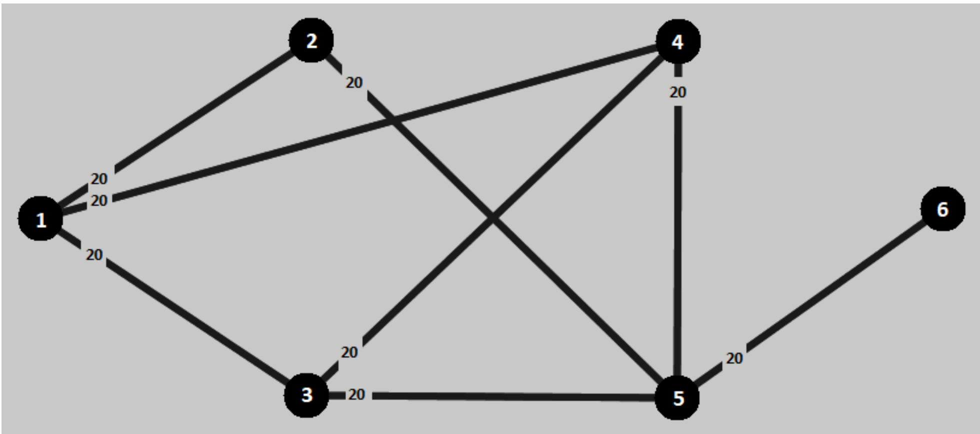

Figura 1.110: ODU2 logical topology defined by the ODU2 traffic matrix.

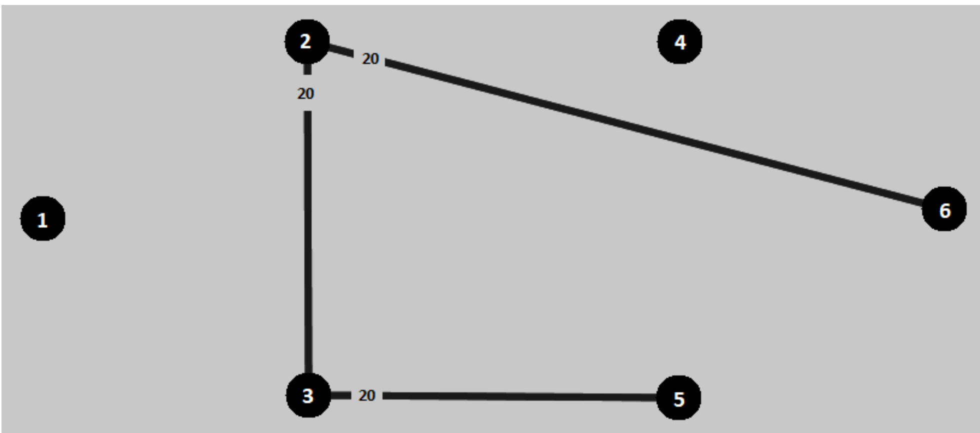

Figura 1.111: ODU3 logical topology defined by the ODU3 traffic matrix.

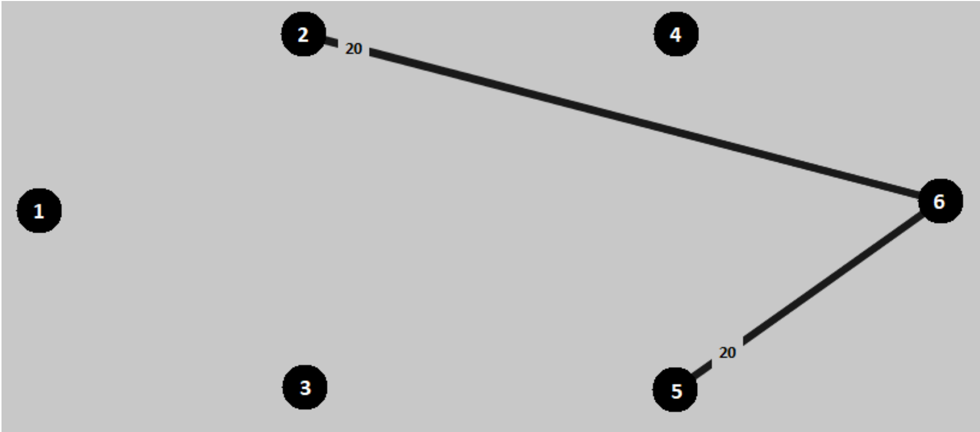

Figura 1.112: ODU4 logical topology defined by the ODU4 traffic matrix.

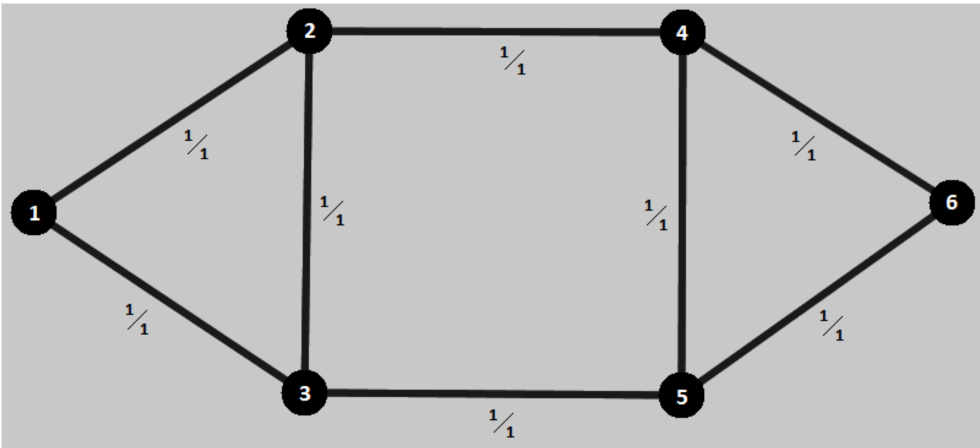

Figura 1.113: Physical topology after dimensioning.

Following all the steps mentioned in the **??**, applying the routing and grooming heuristic algorithms in the Net2Plan software and using all the data referring to this scenario, the obtained result for the Vasco's heuristics can be consulted in the following table 1.67. In table 1.65 mentioned in previous scenario we can see how all the values were calculated.

CAPEX of the Network						
			Quantity	Unit Price	Cost	Total
Link Cost	OLTs		16	15 000 €	240 000 €	157 520 000 €
	100 Gbits/s Transceivers		314	5 000 €/Gbit/s	157 000 000 €	
	Amplifiers		70	4 000 €	280 000 €	
Node Cost	Electrical	EXCs	6	10 000 €	60 000 €	28 486 800 €
		ODU0 Ports	1 200	10 €/port	12 000 €	
		ODU1 Ports	1 000	15 €/port	15 000 €	
		ODU2 Ports	320	30 €/port	9 600 €	
		ODU3 Ports	120	60 €/port	7 200 €	
		ODU4 Ports	80	100 €/port	8 000 €	
		Transponders	268	100 000 €/port	26 800 000 €	
	Optical	OXCs	6	20 000 €	120 000 €	
		Line Ports	314	2 500 €/port	785 000 €	
		Add Ports	268	2 500 €/port	670 000 €	
Total Network Cost						186 006 800 €

Tabela 1.67: Table with detailed description of CAPEX of Vasco's 2016 results.

Conclusions

Once we have obtained the results for all the scenarios we will now draw some conclusions about these results. For a better analysis of the results will be created the table 1.68 with the number of line ports, tributary ports and transceivers because they are important values for the cost of CAPEX, the cost of links, the cost of nodes and finally the cost of CAPEX.

	Low Traffic	Medium Traffic	High Traffic
Traffic (Gbit/s)	500	5 000	10 000
Bidirectional Links used	8	8	8
Number of Add ports	34	142	268
Number of Line ports	52	168	314
Number of Tributary ports	136	1 360	2 720
Number of Transceivers	52	168	314
Link Cost	26 520 000 €	84 520 000 €	157 520 000 €
Node Cost	3 797 590 €	15 180 900 €	28 486 800 €
CAPEX	**30 317 590 €**	**99 700 900 €**	**186 006 800 €**
CAPEX/Gbit/s	**60 635 €/Gbit/s**	**19 940 €/Gbit/s**	**18 600 €/Gbit/s**

Tabela 1.68: Table with different value of CAPEX for this case.

Looking at the previous table we can make some comparisons between the several scenario:

- Comparing the low traffic with the others we can see that despite having an increase of factor ten (medium traffic) and factor twenty (high traffic), the same increase does not occur in the final cost (it is lower);

  This happens because the number of the transceivers is lower than expected which leads by carrying the traffic with less network components and, consequently, the network CAPEX is lower;

- Comparing the medium traffic with the high traffic we can see that the increase of the factor is double and in the final cost this factor is very close but still inferior;

  This happens because the number of the transceivers is also lower but very close to the expected;

- Comparing the CAPEX cost per bit we can see that in the low traffic the cost is higher than the medium and high traffic, which in these two cases the value is similar, but still inferior in the higher traffic;

  This happens because the lower the traffic, the higher CAPEX/bit will be. We can see that in medium and high traffic the results tend to be one closer and lower value.

Opens Issues

The creation of this model for any scenario, started with some considerations and some open issues being:

- Allow blocking.

  The presented model assume that the solution is possible or impossible, does not support a partial solution where some demands are not routed (are blocked);

- Allow multiple transmission system.

  The presented model for each link only supports one transmission system.

Capítulo 2

Transparent transport mode

Student Name	:	Eduardo Fernandes (09/11/2018 -)
Goal	:	Allows blocking, i.e., creation of a model that supports a partial solution where some demands are not routed (are blocked).

In figure 1.112 a top level diagram is presented in which it is represented the heuristcs approach implemented behind the developed algorithms.

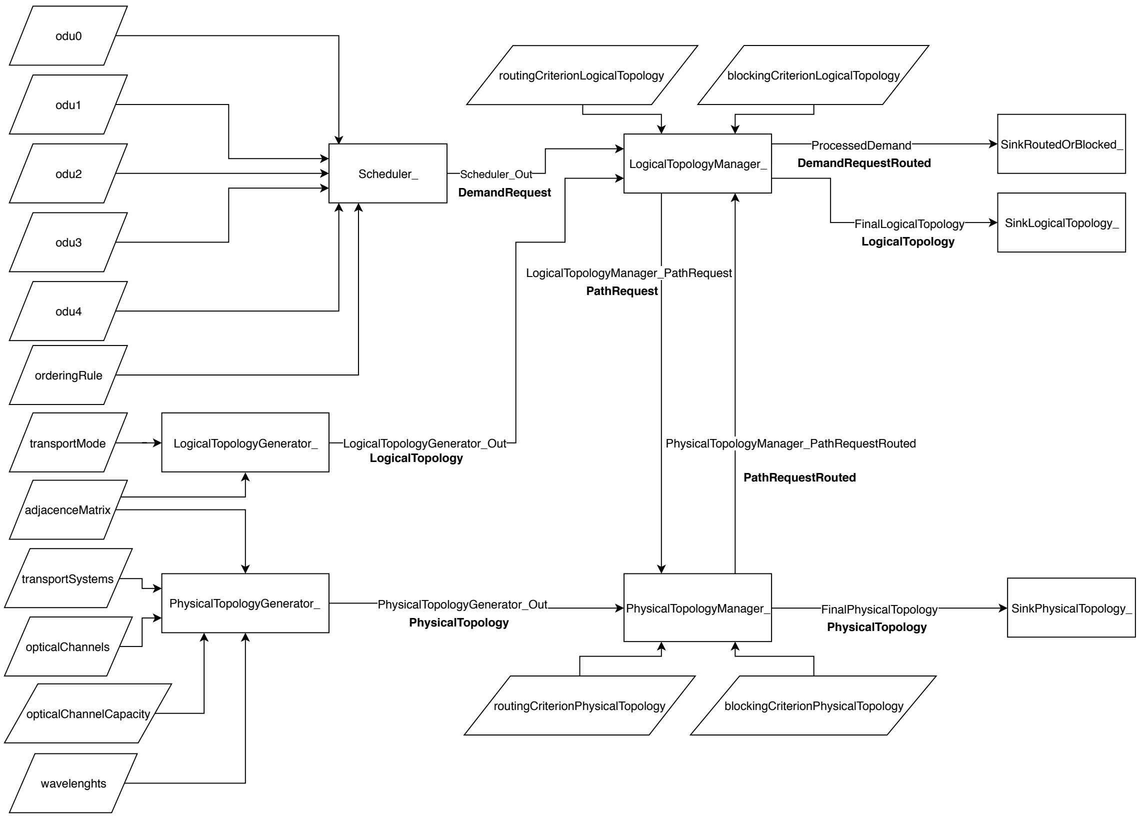

Figura 2.1: High level diagram of the heuristic algorithm performed.

2.1 Concepts

This section serves to introduce some of the terminology used throughout the dissertation.

2.1.1 Network Nodes

The circles in the network graphs presented above represent nodes, these are points in the network that terminate and switch traffic.

2.1.2 Network Links

The lines connecting network nodes can be mentioned both as physical links or optical multiplexing systems and they are tipically depicted with just a single line and are formed by one fiber that carries traffic in a certain direction.

2.2 System inputs

Parameter	Default value	Description
odu0	[0]	N by N matrix containing ODU0 demands.
odu1	[0]	N by N matrix containing ODU1 demands.
odu2	[0]	N by N matrix containing ODU2 demands.
odu3	[0]	N by N matrix containing ODU3 demands.
odu4	[0]	N by N matrix containing ODU4 demands.
orderingRule	descendingOrder	descendingOrder (ODU4 to ODU0) ascendingOrder (ODU0 to ODU4)
transportMode	transparent	"opaque", "transparent"or "translucent".
adjacenceMatrix	[0]	Physical connections of the network.
transportSystems	1 or 2 ?	Number of optical multiplexing systems connecting each pair of nodes.
opticalChannels	100	Optical channels per transport system.
opticalChannelCapacity	100 Gbps	Physical capacity of each optical channel.
routingCriterionLogicalTopology	Hops	Shortest path type.
routingCriterionPhysicalTopology	Hops	Shortest path type.
blockingCriterionLogicalTopology	3	Number of attempted paths before blocking a demand.
blockingCriterionPhysicalTopology	3	Number of attempts before discarding a possible path.
wavelenghts	[0]	Values of wavelengths used (nm).

Tabela 2.1: System input parameters.

2.3 System signals

Signal name	Signal type
Scheduler_Out	DemandRequest
LogicalTopologyGenerator_Out	LogicalTopology
PhysicalTopologyGenerator_Out	PhysicalTopology
LogicalTopologyManager_PathRequest	PathRequest
PhysicalTopologyManager_PathRequestRouted	PathRequestRouted
ProcessedDemand	DemandRequestRouted
FinalLogicalTopology	LogicalTopology
FinalPhysicalTopology	PhysicalTopology

Tabela 2.2: System Signals.

Table 2.2 presents the system signals as well as their type.

2.4 Type signals structures

2.4.1 LogicalTopology

The logical topology of the network is an approach that defines how components are connected. Each node may be optical directly connected to each other, or only optical connected to adjacent nodes or optical connected to suitable nodes. These possibilities are demostrated below on table **??**, where 0/1 indicates the possibility of establishing a logical connection between nodes.

Below are represented the variables that constitute a LogicalTopology type signal.

logicalTopologyMatrix

Nodes	1	2	...	N
1	**0**	0/1	...	0/1
2	0/1	**0**	...	0/1
⋮	⋮	⋮	⋱	⋮
N	0/1	0/1	...	**0**

Tabela 2.3: Allowed logical topology for a matrix of N nodes.

N represents the number of nodes present in the network.

Therefore, these shorter optical paths along the route imposed by logical topology lead to a situation of three possible transport modes: Opaque, Transparent and Translucent. During this dissertation the focus will be on transparent models, where each node connects to all others creating direct links between all nodes of the network.

The next variables are distributed in an hierarchical mode, a Path consists in one or more Light Paths which in its turn are formed by one or more Optical Channels.

paths

pathIndex	sourceNode	destinationNode	capacity (ODU0s)	numberOfLightPaths	lightPathsIndex
0 or greater	1...N	1...N	0...80	0 or greater	[lp_1, lp_2, ...]

Tabela 2.4: Structure of a path variable.

lightPaths

lightPathIndex	sourceNode	destinationNode	capacity (ODU0s)	numberOfOptical-Channels	opticalChannels-Index
0 or greater	1...N	1...N	0..80	0 or greater	[och_1, och_2,...]

Tabela 2.5: Structure of a light path variable.

opticalChannels

opticalChannel-Index	sourceNode	destination-Node	capacity (ODU0s)	wavelenght (nm)	numberOf-Demands	demands-Index
0 or greater	1...N	1...N	0...80	[to define]	0 or greater	[d_1, ...]

Tabela 2.6: Structure of an optical channel variable.

If the network starts with no initial information or files given then these three last variables will be created and sent to the LogicalTopologyManager_ block void, where later on will be updated.

2.4.2 PhysicalTopology

The physical topology can be seen as a layout of a real optical network. The allowed physical topology is defined by the duct and sites in the field. It is assumed that each duct supports up to 2 unidirectional optical multiplexing systems, that will behave like a bidirectional connection between a pair of nodes, and each site supports up to 1 node.

Below are represented the variables that constitute a PhysicalTopology type signal.

physicalTopologyMatrix

Nodes	1	2	...	N
1	**0**	0/1	...	0/1
2	0/1	**0**	...	0/1
⋮	⋮	⋮	⋱	⋮
N	0/1	0/1	...	**0**

Tabela 2.7: Allowed physical topology for a matrix of N nodes.

0/1 indicates the possibility of existing a physical connection between nodes.

Based on the previous adjacence matrix a data structure is created for each of the unidirectional optical multiplexing systems interconnecting nodes.

opticalMultiplexingSystems

opticalMultiplexing-SystemIndex	sourceNode	destination-Node	numberOf-Wavelenghts	wavelenghts	available-Wavelenghts
0	1...N	1...N	OC	[w_1, w_2, ...]	[0/1 ... 0/1]
...	...	...	...	...	...
L-1	1...N	1...N	OC	[w_1, w_2, ...]	[0/1 ... 0/1]

Tabela 2.8: Structure of the opticalMultiplexingSystems variables.

OC represents the number of optical channels present in a physical link.
L represent the total number of existent unidirectional physical links in the network.
w_1 and w_2 represent values of possible wavelengths to be used.
In the availableWavelenghts array 0/1 dictates if the corresponding wavelenght value of wavelenghts array is available or not to be used, 1 and 0 respectively.

All Optical Multiplexing Systems existent in the network are created in the PhysicalTopologyGenerator_ block. They all contain an index in order to be identified all the time, source and destination nodes, numberOfWavelenghts which translates in capacity in terms of quantity of available wavelenghts, and finally the actual values of wavelenghts and a matrix wich specifies which of them are still available to be used, initially all.

2.4.3 DemandRequest

demandIndex	**sourceNode**	**destinationNode**	**oduType**	**survivabilityMethod**
0...D-1	1...N	1...N	0...4	none protection_1_plus_1 restoration

Tabela 2.9: Constitution of a DemandRequest type signal.

D represents the total number of demand requests entering the network. It is possible to know this value for the fact that we are dealing with static traffic and so all the traffic requests are known. In the situation where all requests for traffic are not known, this traffic is said to be dynamic.

2.4.4 PathRequest

requestIndex	sourceNode	intermediateNodes	destinationNode
0 or greater	1...N	1...N	1...N

Tabela 2.10: PathRequest type signal.

The PathRequest type signal will be sent from the LogicalTopologyManager_ block into the PhysicalTopologyManager_ block asking for a path to be created between source and destination nodes of a certain demand. In order establish that path one or more light paths are required. In this specific case, transparent transport mode, only direct logical connections will be taken into account, because the information has to travel only in the optical domain from source to destination, and so all paths created will be formed by only one direct light path. This means that there will be no intermediate nodes.

2.4.5 PathRequestRouted

A PathRequestRouted type signal is formed the following two structures, pathInformation and lightPathsTable, that will inform the LogicalTopologyManager_ block about the possibility of establishing the path requested.

pathInformation

requestIndex	routed	numberOfLightPaths
0 or greater	true/false	0 or greater

Tabela 2.11: pathInformation variable structure.

lightPathsTable

sourceNode	destinationNode	numberOfIntermidiateNodes	intermediateNodes	wavelenght
1...N	1...N	0...N-2	[1...N ... 1...N]	W
...	...	...	...	...

Tabela 2.12: lightPathsTable variable structure.

The lightPathsTable structure will numberOfLightPaths-1 elements.

This signal represents the response that the PhysicalTopologyManager_ block sends back to the LogicalTopologyManager_ block when asked to establish a path. There is an requestIndex variable that identifies this signal as a reponse to the pathRequest signal with the same requestIndex. It is also formed by one boolean variable "routed"which will return true in the case a demand is routed correctly through the network, validating the remaining information on the lighPathsTable, or false in the case it is not, which means no path is possible to be created to route the demand and so the other fields of this signal will be void or fill with invalid information. Furthermore, this signal contains, in the lightPathsTable, various information about each of the light paths needed to establish the path required in the case it is possible to do so.

2.4.6 DemandRequestRouted

demandIndex	routed	pathIndex
0...D-1	true/false	0 or greater

Tabela 2.13: DemandRequestRouted type signal.

This signal contains information regarding one demand and whether it was routed or not. In the case variable "routed"assumes a true value it is presented in the final block (SinkRoutedOrBlocked_) of the diagram of figure 2.1 information about the path used. In the case where variable "routed"assumes value false only the information about the demandIndex is presented meaning that that demand was blocked. In the case a demand is routed combining the information of the DemandRequestRouted signal with the one present in the LogicalTopologyManager_ block it is possible to know every piece of information about the route taken by the demand, lighPaths and opticalChannels used as well as waveleghts.

2.5 Blocks input parameters and signals

Block	Input Parameters	Input Signals
Scheduler_	odu0, odu1, odu2, odu3, odu4, orderingRule	None
LogicalTopologyGenerator_	transportMode, adjacenceMatrix	None
PhysicalTopologyGenerator_	transportSystems, opticalChannels, opticalChannelCapacity	None
LogicalTopologyManager_	routingCriterionLogicalTopology, blockingCriterionLogicalTopology	Scheduler_Out, LogicalTopologyGenerator_Out, PhysicalTopologyManagerOut
PhysicalTopologyManager_	routingCriterionPhysicalTopology, blockingCriterionPhysicalTopology	PhysicalGeneratorOut, LogicalManagerRequest
SinkRoutedOrBlocked_	None	LogicalManagerOut
SinkLogicalTopology_	None	currentLogicalTopology
SinkPhysicalTopology_	None	currentPhysicalTopology

Tabela 2.14: Blocks input parameters and signals.

2.6 Blocks state variables and output signals

Block	State Variables	Output Signals
Scheduler_	odu0, odu1, odu2, odu3, odu4, demandIndex, numberOfDemands	SchedulerOut
LogicalTopologyGenerator_	generate	LogicalTopologyOut
PhysicalTopologyGenerator_	generate	PhysicalTopologyOut
LogicalTopologyManager_	paths, lightPaths, opticalChannels	LogicalManagerRequest, LogicalManagerOut
PhysicalTopologyManager_	opticalMultiplexingSystems	PhysicalManagerOut
SinkRoutedOrBlocked_	None	None

Tabela 2.15: Blocks state variables and output signals.

2.7 Network example

2.7.1 Physical topology matrix

Considering a network with the following adjacence matrix represented on table 2.50, it becomes evident that our test network consists in 6 nodes and 16 unidirectional optical multiplexing systems.

Nodes	1	2	3	4	5	6
1	**0**	1	0	0	0	1
2	1	**0**	1	0	0	1
3	0	1	**0**	1	1	0
4	0	0	1	**0**	1	0
5	0	0	1	1	**0**	1
6	1	1	0	0	1	**0**

Tabela 2.16: Allowed physical topology matrix.

Below we have a graphical representation of the same network physical topology. Each arrow represents an optical multiplexing system connecting two nodes in a certain direction and all of them have a number which reprents its own index for further identification purposes.

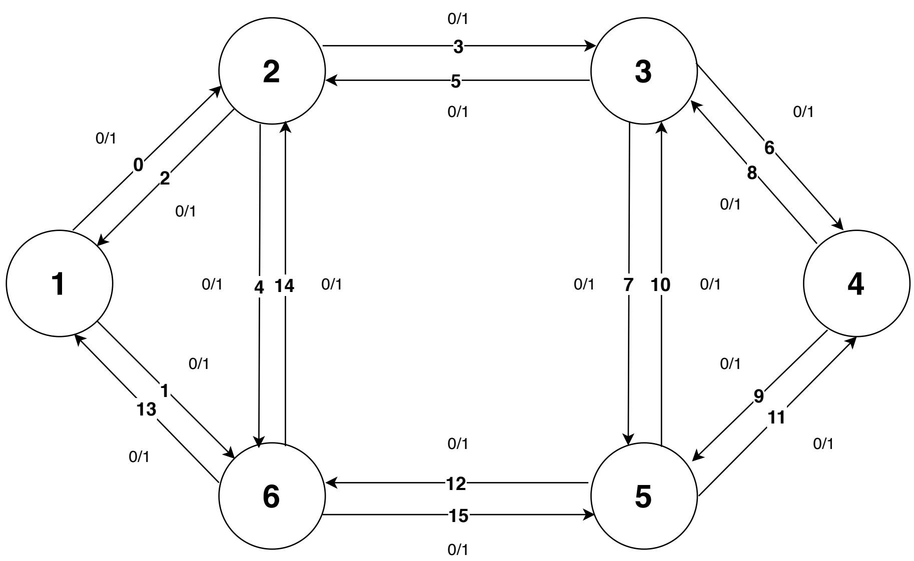

Figura 2.2: Allowed physical topology.

2.7.2 Logical topology matrix

The network logical topology represents how the links and the nodes may interconnect with each other. In this particular case it is considered the transparent transport mode, and so each of the nodes will connect to others creating direct links between all nodes of the network. The same can be seen below represented in a matrix in table 2.17 and graphically in figure 2.3.

Nodes	1	2	3	4	5	6
1	**0**	1	1	1	1	1
2	1	**0**	1	1	1	1
3	1	1	**0**	1	1	1
4	1	1	1	**0**	1	1
5	1	1	1	1	**0**	1
6	1	1	1	1	1	**0**

Tabela 2.17: Allowed logical topology matrix for transparent transport mode.

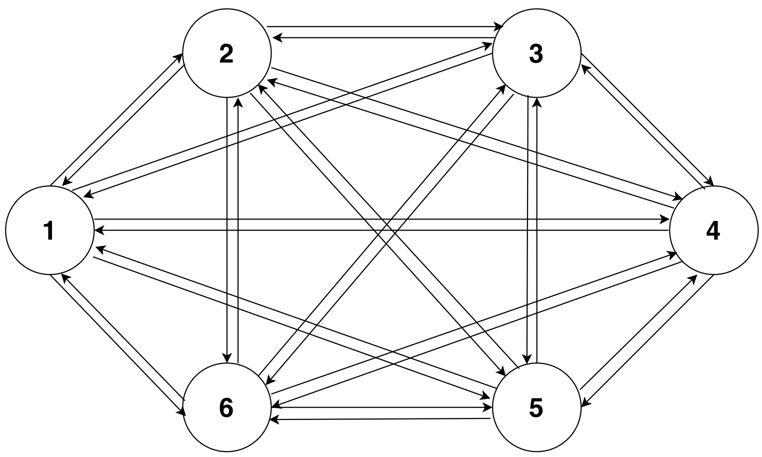

Figura 2.3: Allowed logical topology graph.

2.7.3 Traffic scenario

The traffic matrices below are represented by ODU0, ODU1, ODU2, ODU3 and ODU4 where each one has a certain bit rate. The ODU0 corresponds to 1.25 Gbits/s, the ODU1 corresponds to 2.5 Gbits/s, the ODU2 corresponds to 10 Gbits/s, the ODU3 corresponds to 40 Gbits/s and finally the ODU4 corresponds to 100 Gbits/s.

Considering just 2 optical channels per opticalMultiplexingSystem, each with a capacity of 100 Gbps (80 ODU0s), and the following set of demands.

$$ODU0 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad ODU1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$ODU2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad ODU3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$ODU4 = \begin{bmatrix} 0 & 0 & 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

2.7.4 Theorectical resolution

Through these ODU's we can calculate the total network traffic for this specific scenario:

T_1^0 = 0x1.25 = 0 Gbits/s T_1^1 = 0x2.5 = 0 Gbits/s T_1^2 = 0x10 = 0 Gbits/s

T_1^3 = 2x40 = 80 Gbits/s T_1^4 = 5x100 = 500 Gbits/s

T_1 = 0 + 0 + 0 + 80 + 500 = 580 Gbits/s

Where the variable T_1^x represents the unidirectional traffic of the ODUx, for example, T_1^0 represents the unidirectional traffic of the ODU0 and T_1^1 represents the unidirectional traffic of the ODU1. The variable T_1 represents the total of unidirectional traffic that is injected into the network.

Ordering demands

The demands are first ordered in the Scheduler_ block considering the entry variable, orderingRule, in a single queue. It is assumed orderingRule to be equal to its default value descendingOrder, which will organize demand requests from ODU4 type demands to ODU0 type. These demands will constitute the Scheduler_Out signal, which is a DemandRequest type signal.

demandIndex	sourceNode	destinationNode	oduType	survivabilityMethod
0	1	5	4	none
1	1	5	4	none
2	1	5	4	none
3	1	5	4	none
4	1	5	4	none
5	3	5	3	none
6	3	5	3	none

Tabela 2.18: Demands ordered by Scheduler_ block.

Demand 0

Generate a DemandRequest type signal

Demand 0 is sent in a DemandRequest type signal from the Scheduler_ block to the LogicalTopologyManager_ block.

demandIndex	sourceNode	destinationNode	oduType	survivabilityMethod
0	1	5	4	none

Tabela 2.19: DemandRequest sent by Scheduler_ block for demand 0.

There, a search is made by an available path from source to destination node, in this case nodes 1 and 5, respectively.

Search for an availabe path

pathIndex	sourceNode	destinationNode	capacity (ODU0s)	lightPathsIndex

Tabela 2.20: Paths state variable from LogicalTopologyManager_ block.

As initially no path exists, one must be created in order to route this demand and so a pathRequest type signal is sent from the LogicalTopologyManager_ to to the PhysicalTopologyManager_ block.

Apply Dijkstra algorithm to discover shortest logical paths

The logical topology matrix of table 2.17 is applied as an entry parameter to the Dijkstra algorithm. Considering the entry variables routingCriterionLogicalTopology as "Hops"(gives the shortest paths in terms of hops) and blockingCriterionLogicalTopology equal to 3, once in transparent transport mode only direct logical connections will concerne us, the output gives only the possibility of the direct logical connection (1->5) to establish a path between source and destination nodes, 1 and 5 respectively.

Generate a pathRequest signal

requestIndex	sourceNode	intermidiateNodes	destinationNode
0	1	-1	5

Tabela 2.21: pathRequest signal for demand 0.

As the transport mode assumed is the transparent, only direct logical connections are used to route demands, and consequently there are no intermidiate nodes.

Apply Dijkstra algorithm to discover shortest physical paths

The physical topology matrix of table 2.50 is applied as an entry parameter to the Dijkstra algorithm. Here it is intended to convert the required light path (1->5) into a restricted variety of sets of physical connections, in this specific case 3. Considering blockingCriterionPhysicalTopology equal to 3 (number of paths to be tested before declaring a possible blocked state) and routingCriterionPhysicalTopology to be "Hops", then the final result of Dijkstra algorithm will be the one presented below.

	lightPath from nodes 1 to 5			
	sourceNode	destinationNode	opticalChannels	
			sourceNode	destinationNode
Shortest path 1	1	5	1	6
			6	5
Shortest path 2	1	5	1	2
			2	3
			3	5
Shortest path 3	1	5	1	2
			2	6
			6	5

Tabela 2.22: Result of Dijkstra algorithm applied to the ajacence matrix.

Test shortest path 1

opticalMultiplexing-SystemIndex	sourceNode	destination-Node	numberOf-Wavelenghts	wavelenghts (nm)	available-Wavelenghts
1	1	6	2	[1310 1550]	[1 1]
15	6	5	2	[1310 1550]	[1 1]

Tabela 2.23: Initial values of opticalMultiplexingSystems going from node 1 to 6 and 6 to 5.

As both opticalMultiplexingSystems have the required capacity and wavelenghts a PathRequestRouted signal will be generated informing the LogicalTopologyManager_ block that it is possible to establish a path to route demand 0 through a lightPath using the opticalChannels above, and the physical capacity of the opticalMultiplexingSystems of the network will be updated.

opticalMultiplexing-SystemIndex	sourceNode	destination-Node	numberOf-Wavelenghts	wavelenghts (nm)	available-Wavelenghts
1	1	6	2	[1310 1550]	[0 1]
15	6	5	2	[1310 1550]	[0 1]

Tabela 2.24: After values of opticalMultiplexingSystems going from node 1 to 6 and 6 to 5.

In both Optical Multiplexing Systems an Optical Channel is going to be used in the creation of a Light Path in order to route demand 0. In this specific case the wavelenght used is 1310 nm which will remain unavailable for other demands that will not use this Light Path.

Generate a PathRequestRouted type signal

requestIndex	routed	numberOfLightPaths
0	true	1

Tabela 2.25: pathInformation variable from PathRequestRouted signal.

sourceNode	destinationNode	numberOfIntermediateNodes	intermediateNodes	wavelenght (nm)
1	5	1	[6]	1310

Tabela 2.26: lightPathsTable variable from PathRequestRouted signal.

Update LogicalTopologyManager_ block state variables

pathIndex	sourceNode	destinationNodes	capacity (ODU0s)	numberOfLightPaths	lightPathsIndex
0	1	5	0	1	0

Tabela 2.27: Paths variable updated.

lightPathIndex	sourceNode	destinationNode	capacity (ODU0s)	numberOfOptical-Channels	opticalChannles-Index
0	1	5	0	2	[0,1]

Tabela 2.28: LightPaths variable updated.

opticalChannel-Index	sourceNode	destination-Node	capacity (ODU0s)	wavelenght (nm)	numberOf-Demands	demands-Index
0	1	6	0	1310	1	[0]
1	6	5	0	1310	1	[0]

Tabela 2.29: OpticalChannels variable updated.

Nodes	1	2	3	4	5	6
1	**0**	1	1	1	1	1
2	1	**0**	1	1	1	1
3	1	1	**0**	1	1	1
4	1	1	1	**0**	1	1
5	1	1	1	1	**0**	1
6	1	1	1	1	1	**0**

Tabela 2.30: logicalTopologyMatrix variable updated.

Generate a DemandRequestRouted type signal

demandIndex	routed	pathsIndex
0	true	0

Tabela 2.31: DemandRequestRouted signal.

Finally this signal is sent to the last block of the diagram, SinkRoutedOrBlocked_.

2.7.5 Demand 1

Generate a DemandRequest type signal

Demand 1 is sent in a DemandRequest type signal from the Scheduler_ block to the LogicalTopologyManager_ block.

demandIndex	sourceNode	destinationNode	oduType	survivabilityMethod
1	1	5	4	none

Tabela 2.32: DemandRequest sent by Scheduler_ block for demand 1.

There, a search is made by an available path from source to destination node, in this case nodes 1 and 5, respectively.

Search for an availabe path

pathIndex	sourceNode	destinationNodes	capacity (ODU0s)	numberOfLightPaths	lightPathsIndex
0	1	5	0	1	0

Tabela 2.33: Paths state variable from LogicalTopologyManager_ block.

As there is no path available with remaning capacity to route an ODU4 demand, one must be created in order to route this demand 1 and so a pathRequest type signal is sent from the LogicalTopologyManager_ to to the PhysicalTopologyManager_ block.

Apply Dijkstra algorithm to discover shortest logical paths

The logical topology matrix of table 2.110 is applied as an entry parameter to the Dijkstra algorithm. Considering the entry variables routingCriterionLogicalTopology as "Hops"(gives the shortest paths in terms of hops) and blockingCriterionLogicalTopology equal to 3, once in transparent transport mode only direct logical connections will concerne us, the output gives only the possibility of the direct logical connection (1->5) to establish a path between source and destination nodes, 1 and 5 respectively.

Generate a pathRequest signal

requestIndex	sourceNode	intermidiateNodes	destinationNode
1	1	-1	5

Tabela 2.34: pathRequest signal for demand 0.

As the transport mode assumed is the transparent, only direct logical connections are used to route demands, and consequently there are no intermidiate nodes.

Apply Dijkstra algorithm to discover shortest physical paths

The physical topology matrix of table 2.50 is applied as an entry parameter to the Dijkstra algorithm. Here it is intended to convert the required light path (1->5) into a restricted variety of sets of physical connections, in this specific case 3. Considering blockingCriterionPhysicalTopology equal to 3 (number of paths to be tested before declaring a possible blocked state) and routingCriterionPhysicalTopology to be "Hops", then the final result of Dijkstra algorithm will be the one presented below.

	lightPath from nodes 1 to 5			
	sourceNode	destinationNode	opticalChannels	
			sourceNode	destinationNode
Shortest path 1	1	5	1	6
			6	5
Shortest path 2	1	5	1	2
			2	3
			3	5
Shortest path 3	1	5	1	2
			2	6
			6	5

Tabela 2.35: Result of Dijkstra algorithm applied to the ajacence matrix.

Test shortest path 1

opticalMultiplexing-SystemIndex	sourceNode	destination-Node	numberOf-Wavelenghts	wavelenghts (nm)	available-Wavelenghts
1	1	6	2	[1310 1550]	[0 1]
15	6	5	2	[1310 1550]	[0 1]

Tabela 2.36: Initial values of opticalMultiplexingSystems going from node 1 to 6 and 6 to 5.

As both opticalMultiplexingSystems have the required capacity and wavelenghts a PathRequestRouted signal will be generated informing the LogicalTopologyManager_ block that it is possible to establish a path to route demand 1 through a lightPath using the opticalChannels above, and the physical capacity of the opticalMultiplexingSystems of the network will be updated.

opticalMultiplexing-SystemIndex	sourceNode	destination-Node	numberOf-Wavelenghts	wavelenghts (nm)	available-Wavelenghts
1	1	6	2	[1310 1550]	[0 0]
15	6	5	2	[1310 1550]	[0 0]

Tabela 2.37: After values of opticalMultiplexingSystems going from node 1 to 6 and 6 to 5.

In both Optical Multiplexing Systems an Optical Channel is going to be used in the creation of a Light Path in order to route demand 1. In this specific case the wavelenght used is 1550 nm which will remain unavailable for other demands that will not use this Light Path.

Generate a PathRequestRouted type signal

requestIndex	routed	numberOfLightPaths
1	true	1

Tabela 2.38: pathInformation variable from PathRequestRouted signal.

sourceNode	destinationNode	numberOfIntermediateNodes	intermediateNodes	wavelenght (nm)
1	5	1	[6]	1550

Tabela 2.39: lightPathsTable variable from PathRequestRouted signal.

Update LogicalTopologyManager_ block state variables

pathIndex	sourceNode	destinationNodes	capacity (ODU0s)	numberOfLightPaths	lightPathsIndex
0	1	5	0	1	0
1	1	5	0	1	1

Tabela 2.40: Paths variable updated.

lightPathIndex	sourceNode	destinationNode	capacity (ODU0s)	numberOfOptical-Channels	opticalChannles-Index
0	1	5	0	2	[0,1]
1	1	5	0	2	[2,3]

Tabela 2.41: LightPaths variable updated.

opticalChannel-Index	sourceNode	destination-Node	capacity (ODU0s)	wavelenght (nm)	numberOf-Demands	demands-Index
0	1	6	0	1310	1	[0]
1	6	5	0	1310	1	[0]
2	1	6	0	1550	1	[1]
3	6	5	0	1550	1	[1]

Tabela 2.42: OpticalChannels variable updated.

Nodes	1	2	3	4	5	6
1	**0**	1	1	1	1	1
2	1	**0**	1	1	1	1
3	1	1	**0**	1	1	1
4	1	1	1	**0**	1	1
5	1	1	1	1	**0**	1
6	1	1	1	1	1	**0**

Tabela 2.43: logicalTopologyMatrix variable updated.

Generate a DemandRequestRouted type signal

demandIndex	routed	pathsIndex
1	true	1

Tabela 2.44: DemandRequestRouted signal.

Finally this signal is sent to the last block of the diagram, SinkRoutedOrBlocked_.

2.7.6 Demand 2

Generate a DemandRequest type signal

Demand 2 is sent in a DemandRequest type signal from the Scheduler_ block to the LogicalTopologyManager_ block.

demandIndex	sourceNode	destinationNode	oduType	survivabilityMethod
2	1	5	4	none

Tabela 2.45: DemandRequest sent by Scheduler_ block for demand 2.

There, a search is made by an available path from source to destination node, in this case nodes 1 and 5, respectively.

Search for an availabe path

pathIndex	sourceNode	destinationNodes	capacity (ODU0s)	numberOfLightPaths	lightPathsIndex
0	1	5	0	1	0
1	1	5	0	1	1

Tabela 2.46: Paths state variable from LogicalTopologyManager_ block.

As there is no path available with remaning capacity to route an ODU4 demand, one must be created in order to route this demand 2 and so a pathRequest type signal is sent from the LogicalTopologyManager_ to to the PhysicalTopologyManager_ block.

Apply Dijkstra algorithm to discover shortest logical paths

The logical topology matrix of table 2.110 is applied as an entry parameter to the Dijkstra algorithm. Considering the entry variables routingCriterionLogicalTopology as "Hops"(gives the shortest paths in terms of hops) and blockingCriterionLogicalTopology equal to 3, once in transparent transport mode only direct logical connections will concerne us, the output gives only the possibility of the direct logical connection (1->5) to establish a path between source and destination nodes, 1 and 5 respectively.

Generate a pathRequest signal

requestIndex	sourceNode	intermidiateNodes	destinationNode
2	1	-1	5

Tabela 2.47: pathRequest signal for demand 0.

As the transport mode assumed is the transparent, only direct logical connections are used to route demands, and consequently there are no intermidiate nodes.

Apply Dijkstra algorithm to discover shortest physical paths

The physical topology matrix of table 2.50 is applied as an entry parameter to the Dijkstra algorithm. Here it is intended to convert the required light path (1->5) into a restricted variety of sets of physical connections, in this specific case 3. Considering blockingCriterionPhysicalTopology equal to 3 (number of paths to be tested before declaring a possible blocked state) and routingCriterionPhysicalTopology to be "Hops", then the final result of Dijkstra algorithm will be the one presented below.

	lightPath from nodes 1 to 5			
	sourceNode	destinationNode	opticalChannels	
			sourceNode	destinationNode
Shortest path 1	1	5	1	6
			6	5
Shortest path 2	1	5	1	2
			2	3
			3	5
Shortest path 3	1	5	1	2
			2	6
			6	5

Tabela 2.48: Result of Dijkstra algorithm applied to the ajacence matrix.

Test shortest path 1

opticalMultiplexing-SystemIndex	sourceNode	destination-Node	numberOf-Wavelenghts	wavelenghts (nm)	available-Wavelenghts
1	1	6	2	[1310 1550]	[0 0]
15	6	5	2	[1310 1550]	[0 0]

Tabela 2.49: Initial values of opticalMultiplexingSystems going from node 1 to 6 and 6 to 5.

As both opticalMultiplexingSystems are already completely full then they will remain unavailable for demands to use and so the physical topology must be updated so that Dijkstra algorithm does not take into account this Optical Multiplexing Systems for shortest path creating processes in further iterations.

Update physical topology matrix

Nodes	1	2	3	4	5	6
1	**0**	1	0	0	0	0
2	1	**0**	1	0	0	1
3	0	1	**0**	1	1	0
4	0	0	1	**0**	1	0
5	0	0	1	1	**0**	1
6	1	1	0	0	0	**0**

Tabela 2.50: Updated physical topology matrix.

Test shortest path 2

opticalMultiplexing-SystemIndex	sourceNode	destination-Node	numberOf-Wavelenghts	wavelenghts (nm)	available-Wavelenghts
0	1	2	2	[1310 1550]	[1 1]
3	2	3	2	[1310 1550]	[1 1]
7	3	5	2	[1310 1550]	[1 1]

Tabela 2.51: Initial values of opticalMultiplexingSystems going from nodes 1 to 2, 2 to 3 and 3 to 5.

opticalMultiplexing-SystemIndex	sourceNode	destination-Node	numberOf-Wavelenghts	wavelenghts (nm)	available-Wavelenghts
0	1	2	2	[1310 1550]	[0 1]
3	2	3	2	[1310 1550]	[0 1]
7	3	5	2	[1310 1550]	[0 1]

Tabela 2.52: After values of opticalMultiplexingSystems going from nodes 1 to 2, 2 to 3 and 3 to 5.

In both Optical Multiplexing Systems an Optical Channel is going to be used in the creation of a Light Path in order to route demand 2. In this specific case the wavelenght used is 1310 nm which will remain unavailable for other demands that will not use this Light Path.

Generate a PathRequestRouted type signal

requestIndex	routed	numberOfLightPaths
2	true	1

Tabela 2.53: pathInformation variable from PathRequestRouted signal.

sourceNode	destinationNode	numberOfIntermediateNodes	intermediateNodes	wavelenght (nm)
1	5	2	[2 3]	1310

Tabela 2.54: lightPathsTable variable from PathRequestRouted signal.

Update LogicalTopologyManager_ block state variables

pathIndex	sourceNode	destinationNodes	capacity (ODU0s)	numberOfLightPaths	lightPathsIndex
0	1	5	0	1	0
1	1	5	0	1	1
2	1	5	0	1	2

Tabela 2.55: Paths variable updated.

lightPathIndex	sourceNode	destinationNode	capacity (ODU0s)	numberOfOptical-Channels	opticalChannles-Index
0	1	5	0	2	[0,1]
1	1	5	0	2	[2,3]
2	1	5	0	3	[4,5,6]

Tabela 2.56: LightPaths variable updated.

opticalChannel-Index	sourceNode	destination-Node	capacity (ODU0s)	wavelenght (nm)	numberOf-Demands	demands-Index
0	1	6	0	1310	1	[0]
1	6	5	0	1310	1	[0]
2	1	6	0	1550	1	[1]
3	6	5	0	1550	1	[1]
4	1	2	0	1310	1	[2]
5	2	3	0	1310	1	[2]
6	3	5	0	1310	1	[2]

Tabela 2.57: OpticalChannels variable updated.

Nodes	1	2	3	4	5	6
1	**0**	1	1	1	1	1
2	1	**0**	1	1	1	1
3	1	1	**0**	1	1	1
4	1	1	1	**0**	1	1
5	1	1	1	1	**0**	1
6	1	1	1	1	1	**0**

Tabela 2.58: logicalTopologyMatrix variable updated.

Generate a DemandRequestRouted type signal

demandIndex	routed	pathsIndex
2	true	2

Tabela 2.59: DemandRequestRouted signal.

Finally this signal is sent to the last block of the diagram, SinkRoutedOrBlocked_.

2.7.7 Demand 3

Generate a DemandRequest type signal

Demand 3 is sent in a DemandRequest type signal from the Scheduler_ block to the LogicalTopologyManager_ block.

demandIndex	sourceNode	destinationNode	oduType	survivabilityMethod
3	1	5	4	none

Tabela 2.60: DemandRequest sent by Scheduler_ block for demand 3.

There, a search is made by an available path from source to destination node, in this case nodes 1 and 5, respectively.

Search for an availabe path

pathIndex	sourceNode	destinationNodes	capacity (ODU0s)	numberOfLightPaths	lightPathsIndex
0	1	5	0	1	0
1	1	5	0	1	1
2	1	5	0	1	2

Tabela 2.61: Paths state variable from LogicalTopologyManager_ block.

As there is no path available with remaning capacity to route an ODU4 demand, one must be created in order to route this demand 1 and so a pathRequest type signal is sent from the LogicalTopologyManager_ to to the PhysicalTopologyManager_ block.

Apply Dijkstra algorithm to discover shortest logical paths

The logical topology matrix of table 2.110 is applied as an entry parameter to the Dijkstra algorithm. Considering the entry variables routingCriterionLogicalTopology as "Hops"(gives the shortest paths in terms of hops) and blockingCriterionLogicalTopology equal to 3, once in transparent transport mode only direct logical connections will concerne us, the output gives only the possibility of the direct logical connection (1->5) to establish a path between source and destination nodes, 1 and 5 respectively.

Generate a pathRequest signal

requestIndex	sourceNode	intermidiateNodes	destinationNode
3	1	-1	5

Tabela 2.62: pathRequest signal for demand 0.

As the transport mode assumed is the transparent, only direct logical connections are used to route demands, and consequently there are no intermidiate nodes.

Apply Dijkstra algorithm to discover shortest physical paths

The physical topology matrix already updated before of table xxx is applied as an entry parameter to the Dijkstra algorithm. Here it is intended to convert the required light path (1->5) into a restricted variety of sets of physical connections, in this specific case 3. Considering blockingCriterionPhysicalTopology equal to 3 (number of paths to be tested before declaring a possible blocked state) and routingCriterionPhysicalTopology to be "Hops", then the final result of Dijkstra algorithm will be the one presented below. Notice that once our entry physical topology is now different then the final output of Dijkstra algorithm will also differ.

<table>
<tr><td rowspan="3"></td><td colspan="4">lightPath from nodes 1 to 5</td></tr>
<tr><td rowspan="2">sourceNode</td><td rowspan="2">destinationNode</td><td colspan="2">opticalChannels</td></tr>
<tr><td>sourceNode</td><td>destinationNode</td></tr>
<tr><td rowspan="3">Shortest path 1</td><td rowspan="3">1</td><td rowspan="3">5</td><td>1</td><td>2</td></tr>
<tr><td>2</td><td>3</td></tr>
<tr><td>3</td><td>5</td></tr>
<tr><td rowspan="3">Shortest path 2</td><td rowspan="3">1</td><td rowspan="3">5</td><td>1</td><td>2</td></tr>
<tr><td>2</td><td>6</td></tr>
<tr><td>6</td><td>5</td></tr>
<tr><td rowspan="4">Shortest path 3</td><td rowspan="4">1</td><td rowspan="4">5</td><td>1</td><td>2</td></tr>
<tr><td>2</td><td>3</td></tr>
<tr><td>3</td><td>4</td></tr>
<tr><td>4</td><td>5</td></tr>
</table>

Tabela 2.63: Result of Dijkstra algorithm applied to the ajacence matrix.

Test shortest path 1

opticalMultiplexing-SystemIndex	sourceNode	destination-Node	numberOf-Wavelenghts	wavelenghts (nm)	available-Wavelenghts
0	1	2	2	[1310 1550]	[0 1]
3	2	3	2	[1310 1550]	[0 1]
7	3	5	2	[1310 1550]	[0 1]

Tabela 2.64: Initial values of opticalMultiplexingSystems going from node 1 to 2, 2 to 3 and 3 to 5.

As both opticalMultiplexingSystems have the required capacity and wavelenghts a PathRequestRouted signal will be generated informing the LogicalTopologyManager_ block that it is possible to establish a path to route demand 1 through a lightPath using the opticalChannels above, and the physical capacity of the opticalMultiplexingSystems of the network will be updated.

opticalMultiplexing-SystemIndex	sourceNode	destination-Node	numberOf-Wavelenghts	wavelenghts (nm)	available-Wavelenghts
0	1	2	2	[1310 1550]	[0 0]
3	2	3	2	[1310 1550]	[0 0]
7	3	5	2	[1310 1550]	[0 0]

Tabela 2.65: After values of opticalMultiplexingSystems going from node 1 to 2, 2 to 3 and 3 to 5.

In both Optical Multiplexing Systems an Optical Channel is going to be used in the creation of a Light Path in order to route demand 3. In this specific case the wavelenght used is 1550 nm which will remain unavailable for other demands that will not use this Light Path.

Generate a PathRequestRouted type signal

requestIndex	routed	numberOfLightPaths
3	true	1

Tabela 2.66: pathInformation variable from PathRequestRouted signal.

sourceNode	destinationNode	numberOfIntermediateNodes	intermediateNodes	wavelenght (nm)
1	5	2	[2 3]	1550

Tabela 2.67: lightPathsTable variable from PathRequestRouted signal.

Update LogicalTopologyManager_ block state variables

pathIndex	sourceNode	destinationNodes	capacity (ODU0s)	numberOfLightPaths	lightPathsIndex
0	1	5	0	1	0
1	1	5	0	1	1
2	1	5	0	1	2
3	1	5	0	1	3

Tabela 2.68: Paths variable updated.

lightPathIndex	sourceNode	destinationNode	capacity (ODU0s)	numberOfOptical-Channels	opticalChannles-Index
0	1	5	0	2	[0,1]
1	1	5	0	2	[2,3]
2	1	5	0	3	[4,5,6]
3	1	5	0	3	[7,8,9]

Tabela 2.69: LightPaths variable updated.

opticalChannel-Index	sourceNode	destination-Node	capacity (ODU0s)	wavelenght (nm)	numberOf-Demands	demands-Index
0	1	6	0	1310	1	[0]
1	6	5	0	1310	1	[0]
2	1	6	0	1550	1	[1]
3	6	5	0	1550	1	[1]
4	1	2	0	1310	1	[2]
5	2	3	0	1310	1	[2]
6	3	5	0	1310	1	[2]
7	1	2	0	1550	1	[3]
8	2	3	0	1550	1	[3]
9	3	5	0	1550	1	[3]

Tabela 2.70: OpticalChannels variable updated.

Nodes	1	2	3	4	5	6
1	**0**	1	1	1	1	1
2	1	**0**	1	1	1	1
3	1	1	**0**	1	1	1
4	1	1	1	**0**	1	1
5	1	1	1	1	**0**	1
6	1	1	1	1	1	**0**

Tabela 2.71: logicalTopologyMatrix variable updated.

Generate a DemandRequestRouted type signal

demandIndex	routed	pathsIndex
3	true	3

Tabela 2.72: DemandRequestRouted signal.

Finally this signal is sent to the last block of the diagram, SinkRoutedOrBlocked_.

2.7.8 Demand 4

Generate a DemandRequest type signal

Demand 3 is sent in a DemandRequest type signal from the Scheduler_ block to the LogicalTopologyManager_ block.

demandIndex	sourceNode	destinationNode	oduType	survivabilityMethod
4	1	5	4	none

Tabela 2.73: DemandRequest sent by Scheduler_ block for demand 4.

There, a search is made by an available path from source to destination node, in this case nodes 1 and 5, respectively.

Search for an availabe path

pathIndex	sourceNode	destinationNodes	capacity (ODU0s)	numberOfLightPaths	lightPathsIndex
0	1	5	0	1	0
1	1	5	0	1	1
2	1	5	0	1	2
3	1	5	0	1	3

Tabela 2.74: Paths state variable from LogicalTopologyManager_ block.

As there is no path available with remaning capacity to route an ODU4 demand, one must be created in order to route this demand 4 and so a pathRequest type signal is sent from the LogicalTopologyManager_ to to the PhysicalTopologyManager_ block.

Apply Dijkstra algorithm to discover shortest logical paths

The logical topology matrix of table 2.110 is applied as an entry parameter to the Dijkstra algorithm. Considering the entry variables routingCriterionLogicalTopology as "Hops"(gives the shortest paths in terms of hops) and blockingCriterionLogicalTopology equal to 3, once in transparent transport mode only direct logical connections will concerne us, the output gives only the possibility of the direct logical connection (1->5) to establish a path between source and destination nodes, 1 and 5 respectively.

Generate a pathRequest signal

requestIndex	sourceNode	intermidiateNodes	destinationNode
4	1	-1	5

Tabela 2.75: pathRequest signal for demand 4.

As the transport mode assumed is the transparent, only direct logical connections are used to route demands, and consequently there are no intermidiate nodes.

Apply Dijkstra algorithm to discover shortest physical paths

The physical topology matrix already updated before of table xxx is applied as an entry parameter to the Dijkstra algorithm. Here it is intended to convert the required light path (1->5) into a restricted variety of sets of physical connections, in this specific case 3. Considering blockingCriterionPhysicalTopology equal to 3 (number of paths to be tested before declaring a possible blocked state) and routingCriterionPhysicalTopology to be "Hops", then the final result of Dijkstra algorithm will be the one presented below. Notice that once our entry physical topology is now different then the final output of Dijkstra algorithm will also differ.

	lightPath from nodes 1 to 5			
	sourceNode	destinationNode	opticalChannels	
			sourceNode	destinationNode
Shortest path 1	1	5	1	2
			2	3
			3	5
Shortest path 2	1	5	1	2
			2	6
			6	5
Shortest path 3	1	5	1	2
			2	3
			3	4
			4	5

Tabela 2.76: Result of Dijkstra algorithm applied to the ajacence matrix.

Test shortest path 1

opticalMultiplexing-SystemIndex	sourceNode	destination-Node	numberOf-Wavelenghts	wavelenghts (nm)	available-Wavelenghts
0	1	2	2	[1310 1550]	[0 0]
3	2	3	2	[1310 1550]	[0 0]
7	3	5	2	[1310 1550]	[0 0]

Tabela 2.77: Initial values of opticalMultiplexingSystems going from node 1 to 2, 2 to 3 and 3 to 5.

As all opticalMultiplexingSystems are already completely full then they will remain unavailable for other demands to use and so the physical topology matrix must be updated so that Dijkstra algorithm does not take into account this Optical Multiplexing Systems for shortest path creating processes in further iterations.

Update physical topology matrix

Nodes	1	2	3	4	5	6
1	**0**	0	0	0	0	0
2	1	**0**	0	0	0	1
3	0	1	**0**	1	0	0
4	0	0	1	**0**	1	0
5	0	0	1	1	**0**	1
6	1	1	0	0	0	**0**

Tabela 2.78: Updated physical topology matrix.

Test shortest paths 2 and 3

As both shortest paths share commom Optical Multiplexing Systems with the first then all of them will be discarded and so it will not be possible to create a path between nodes 1 and 5 to route demand 4.

Generate a PathRequestRouted type signal

requestIndex	routed	numberOfLightPaths
4	false	0

Tabela 2.79: pathInformation variable from PathRequestRouted signal.

sourceNode	destinationNode	numberOfIntermediateNodes	intermediateNodes	wavelenght (nm)
			[ ]	

Tabela 2.80: lightPathsTable variable from PathRequestRouted signal.

Update LogicalTopologyManager_ block state variables

pathIndex	sourceNode	destinationNodes	capacity (ODU0s)	numberOfLightPaths	lightPathsIndex
0	1	5	0	1	0
1	1	5	0	1	1
2	1	5	0	1	2
3	1	5	0	1	3

Tabela 2.81: Paths variable updated.

lightPathIndex	sourceNode	destinationNode	capacity (ODU0s)	numberOfOptical-Channels	opticalChannles-Index
0	1	5	0	2	[0,1]
1	1	5	0	2	[2,3]
2	1	5	0	3	[4,5,6]
3	1	5	0	3	[7,8,9]

Tabela 2.82: LightPaths variable updated.

opticalChannel-Index	sourceNode	destination-Node	capacity (ODU0s)	wavelenght (nm)	numberOf-Demands	demands-Index
0	1	6	0	1310	1	[0]
1	6	5	0	1310	1	[0]
2	1	6	0	1550	1	[1]
3	6	5	0	1550	1	[1]
4	1	2	0	1310	1	[2]
5	2	3	0	1310	1	[2]
6	3	5	0	1310	1	[2]
7	1	2	0	1550	1	[3]
8	2	3	0	1550	1	[3]
9	3	5	0	1550	1	[3]

Tabela 2.83: OpticalChannels variable updated.

Nodes	1	2	3	4	5	6
1	**0**	1	1	1	0	1
2	1	**0**	1	1	1	1
3	1	1	**0**	1	1	1
4	1	1	1	**0**	1	1
5	1	1	1	1	**0**	1
6	1	1	1	1	1	**0**

Tabela 2.84: logicalTopologyMatrix variable updated.

Generate a DemandRequestRouted type signal

demandIndex	routed	pathsIndex
4	false	

Tabela 2.85: DemandRequestRouted signal.

Finally this signal is sent to the last block of the diagram, SinkRoutedOrBlocked_.

2.7.9 Demand 5

Generate a DemandRequest type signal

Demand 5 is sent in a DemandRequest type signal from the Scheduler_ block to the LogicalTopologyManager_ block.

demandIndex	sourceNode	destinationNode	oduType	survivabilityMethod
5	3	5	3	none

Tabela 2.86: DemandRequest sent by Scheduler_ block for demand 5.

There, a search is made by an available path from source to destination node, in this case nodes 3 and 5, respectively.

Search for an availabe path

pathIndex	sourceNode	destinationNodes	capacity (ODU0s)	numberOfLightPaths	lightPathsIndex
0	1	5	0	1	0
1	1	5	0	1	1
2	1	5	0	1	2
3	1	5	0	1	3

Tabela 2.87: Paths state variable from LogicalTopologyManager_ block.

As there is no path available to route demand 5 from nodes 3 to 5 one must be created and so a pathRequest type signal is sent from the LogicalTopologyManager_ to to the PhysicalTopologyManager_ block.

Apply Dijkstra algorithm to discover shortest logical paths

The logical topology matrix of table 2.110 is applied as an entry parameter to the Dijkstra algorithm. Considering the entry variables routingCriterionLogicalTopology as "Hops"(gives the shortest paths in terms of hops) and blockingCriterionLogicalTopology equal to 3, once in transparent transport mode only direct logical connections will concerne us, the output gives only the possibility of the direct logical connection (3->5) to establish a path between source and destination nodes, 3 and 5 respectively.

Generate a pathRequest signal

requestIndex	sourceNode	intermidiateNodes	destinationNode
5	3	-1	5

Tabela 2.88: pathRequest signal for demand 5.

As the transport mode assumed is the transparent, only direct logical connections are used to route demands, and consequently there are no intermidiate nodes.

Apply Dijkstra algorithm to discover shortest physical paths

The physical topology matrix already updated before of table xxx is applied as an entry parameter to the Dijkstra algorithm. Here it is intended to convert the required light path (3->5) into a restricted variety of sets of physical connections, in this specific case 3. Considering blockingCriterionPhysicalTopology equal to 3 (number of paths to be tested before declaring a possible blocked state) and routingCriterionPhysicalTopology to be "Hops", then the final result of Dijkstra algorithm will be the one presented below.

<table>
<tr><td rowspan="3"></td><td colspan="4">lightPath from nodes 1 to 5</td></tr>
<tr><td rowspan="2">sourceNode</td><td rowspan="2">destinationNode</td><td colspan="2">opticalChannels</td></tr>
<tr><td>sourceNode</td><td>destinationNode</td></tr>
<tr><td rowspan="2">Shortest path 1</td><td rowspan="2">3</td><td rowspan="2">5</td><td>3</td><td>4</td></tr>
<tr><td>4</td><td>5</td></tr>
</table>

Tabela 2.89: Result of Dijkstra algorithm applied to the physical topology matrix.

Based on our current physical topology matrix only one path is possible to be established between nodes 3 and 5.

Test shortest path 1

opticalMultiplexing-SystemIndex	sourceNode	destination-Node	numberOf-Wavelenghts	wavelenghts (nm)	available-Wavelenghts
6	3	4	2	[1310 1550]	[1 1]
9	4	5	2	[1310 1550]	[1 1]

Tabela 2.90: Initial values of opticalMultiplexingSystems going from node 3 to 4 and 4 to 5.

As both opticalMultiplexingSystems have the required capacity and wavelenghts a PathRequestRouted signal will be generated informing the LogicalTopologyManager_ block that it is possible to establish a path to route demand 5 through a lightPath using the opticalChannels above, and the physical capacity of the opticalMultiplexingSystems of the network will be updated.

opticalMultiplexing-SystemIndex	sourceNode	destination-Node	numberOf-Wavelenghts	wavelenghts (nm)	available-Wavelenghts
6	3	4	2	[1310 1550]	[0 1]
9	4	5	2	[1310 1550]	[0 1]

Tabela 2.91: After values of opticalMultiplexingSystems going from node 3 to 4 and 4 to 5.

In both Optical Multiplexing Systems an Optical Channel is going to be used in the creation of a Light Path in order to route demand 5. In this specific case the wavelenght used is 1310 nm which will remain unavailable for other demands that will not use this Light Path.

Generate a PathRequestRouted type signal

requestIndex	routed	numberOfLightPaths
5	true	1

Tabela 2.92: pathInformation variable from PathRequestRouted signal.

sourceNode	destinationNode	numberOfIntermediateNodes	intermediateNodes	wavelenght (nm)
3	5	1	[4]	1310

Tabela 2.93: lightPathsTable variable from PathRequestRouted signal.

Update LogicalTopologyManager_ block state variables

pathIndex	sourceNode	destinationNodes	capacity (ODU0s)	numberOfLightPaths	lightPathsIndex
0	1	5	0	1	0
1	1	5	0	1	1
2	1	5	0	1	2
3	1	5	0	1	3
4	3	5	48	1	4

Tabela 2.94: Paths variable updated.

lightPathIndex	sourceNode	destinationNode	capacity (ODU0s)	numberOfOptical-Channels	opticalChannles-Index
0	1	5	0	2	[0,1]
1	1	5	0	2	[2,3]
2	1	5	0	3	[4,5,6]
3	1	5	0	3	[7,8,9]
4	3	5	48	2	[10,11]

Tabela 2.95: LightPaths variable updated.

opticalChannel-Index	sourceNode	destination-Node	capacity (ODU0s)	wavelenght (nm)	numberOf-Demands	demands-Index
0	1	6	0	1310	1	[0]
1	6	5	0	1310	1	[0]
2	1	6	0	1550	1	[1]
3	6	5	0	1550	1	[1]
4	1	2	0	1310	1	[2]
5	2	3	0	1310	1	[2]
6	3	5	0	1310	1	[2]
7	1	2	0	1550	1	[3]
8	2	3	0	1550	1	[3]
9	3	5	0	1550	1	[3]
10	3	4	48	1310	1	[5]
11	3	5	48	1310	1	[5]

Tabela 2.96: OpticalChannels variable updated.

Nodes	1	2	3	4	5	6
1	**0**	1	1	1	0	1
2	1	**0**	1	1	1	1
3	1	1	**0**	1	1	1
4	1	1	1	**0**	1	1
5	1	1	1	1	**0**	1
6	1	1	1	1	1	**0**

Tabela 2.97: logicalTopologyMatrix variable updated.

Generate a DemandRequestRouted type signal

demandIndex	routed	pathsIndex
5	true	4

Tabela 2.98: DemandRequestRouted signal.

Finally this signal is sent to the last block of the diagram, SinkRoutedOrBlocked_.

2.7.10 Demand 6

Generate a DemandRequest type signal

Demand 6 is sent in a DemandRequest type signal from the Scheduler_ block to the LogicalTopologyManager_ block.

demandIndex	sourceNode	destinationNode	oduType	survivabilityMethod
6	3	5	3	none

Tabela 2.99: DemandRequest sent by Scheduler_ block for demand 5.

There, a search is made by an available path from source to destination node, in this case nodes 3 and 5, respectively.

Search for an availabe path

pathIndex	sourceNode	destinationNodes	capacity (ODU0s)	numberOfLightPaths	lightPathsIndex
0	1	5	0	1	0
1	1	5	0	1	1
2	1	5	0	1	2
3	1	5	0	1	3
4	3	5	48	1	4

Tabela 2.100: Paths state variable from LogicalTopologyManager_ block.

As path with index equal to 4 has the same source and destination nodes and there are still capacity available to route an ODU3 type demand, this will then be used to route demand 6 through the network. There will be no necessity to create a pathRequest type signal.

Update LogicalTopologyManager_ block state variables

pathIndex	sourceNode	destinationNodes	capacity (ODU0s)	numberOfLightPaths	lightPathsIndex
0	1	5	0	1	0
1	1	5	0	1	1
2	1	5	0	1	2
3	1	5	0	1	3
4	3	5	16	1	4

Tabela 2.101: Paths variable updated.

lightPathIndex	sourceNode	destinationNode	capacity (ODU0s)	numberOfOptical-Channels	opticalChannles-Index
0	1	5	0	2	[0,1]
1	1	5	0	2	[2,3]
2	1	5	0	3	[4,5,6]
3	1	5	0	3	[7,8,9]
4	3	5	16	2	[10,11]

Tabela 2.102: LightPaths variable updated.

opticalChannel-Index	sourceNode	destination-Node	capacity (ODU0s)	wavelenght (nm)	numberOf-Demands	demands-Index
0	1	6	0	1310	1	[0]
1	6	5	0	1310	1	[0]
2	1	6	0	1550	1	[1]
3	6	5	0	1550	1	[1]
4	1	2	0	1310	1	[2]
5	2	3	0	1310	1	[2]
6	3	5	0	1310	1	[2]
7	1	2	0	1550	1	[3]
8	2	3	0	1550	1	[3]
9	3	5	0	1550	1	[3]
10	3	4	16	1310	2	[5,6]
11	3	5	16	1310	2	[5,6]

Tabela 2.103: OpticalChannels variable updated.

Nodes	1	2	3	4	5	6
1	**0**	1	1	1	0	1
2	1	**0**	1	1	1	1
3	1	1	**0**	1	1	1
4	1	1	1	**0**	1	1
5	1	1	1	1	**0**	1
6	1	1	1	1	1	**0**

Tabela 2.104: logicalTopologyMatrix variable updated.

Generate a DemandRequestRouted type signal

demandIndex	routed	pathsIndex
6	true	4

Tabela 2.105: DemandRequestRouted signal.

Finally this signal is sent to the last block of the diagram, SinkRoutedOrBlocked_.

2.7.11 Final Report

demandIndex	routed	pathsIndex
0	true	0
1	true	1
2	true	2
3	true	3
4	false	
5	true	4
6	true	4

Tabela 2.106: Final information stored in SinkRoutedOrBlocked_ block.

Final Logical Topology

pathIndex	sourceNode	destinationNodes	capacity (ODU0s)	numberOfLightPaths	lightPathsIndex
0	1	5	0	1	0
1	1	5	0	1	1
2	1	5	0	1	2
3	1	5	0	1	3
4	3	5	16	1	4

Tabela 2.107: Final Paths variable value.

lightPathIndex	sourceNode	destinationNode	capacity (ODU0s)	numberOfOptical-Channels	opticalChannles-Index
0	1	5	0	2	[0,1]
1	1	5	0	2	[2,3]
2	1	5	0	3	[4,5,6]
3	1	5	0	3	[7,8,9]
4	3	5	16	2	[10,11]

Tabela 2.108: Final LightPaths variable value.

opticalChannel-Index	sourceNode	destination-Node	capacity (ODU0s)	wavelenght (nm)	numberOf-Demands	demands-Index
0	1	6	0	1310	1	[0]
1	6	5	0	1310	1	[0]
2	1	6	0	1550	1	[1]
3	6	5	0	1550	1	[1]
4	1	2	0	1310	1	[2]
5	2	3	0	1310	1	[2]
6	3	5	0	1310	1	[2]
7	1	2	0	1550	1	[3]
8	2	3	0	1550	1	[3]
9	3	5	0	1550	1	[3]
10	3	4	16	1310	2	[5,6]
11	3	5	16	1310	2	[5,6]

Tabela 2.109: Final OpticalChannels variable value.

Nodes	1	2	3	4	5	6
1	**0**	1	1	1	0	1
2	1	**0**	1	1	1	1
3	1	1	**0**	1	1	1
4	1	1	1	**0**	1	1
5	1	1	1	1	**0**	1
6	1	1	1	1	1	**0**

Tabela 2.110: Final logical topology matrix value.

Final Physical Topology

Nodes	1	2	3	4	5	6
1	**0**	0	0	0	0	0
2	1	**0**	0	0	0	1
3	0	1	**0**	1	0	0
4	0	0	1	**0**	1	0
5	0	0	1	1	**0**	1
6	1	1	0	0	0	**0**

Tabela 2.111: Final physical topology matrix value.

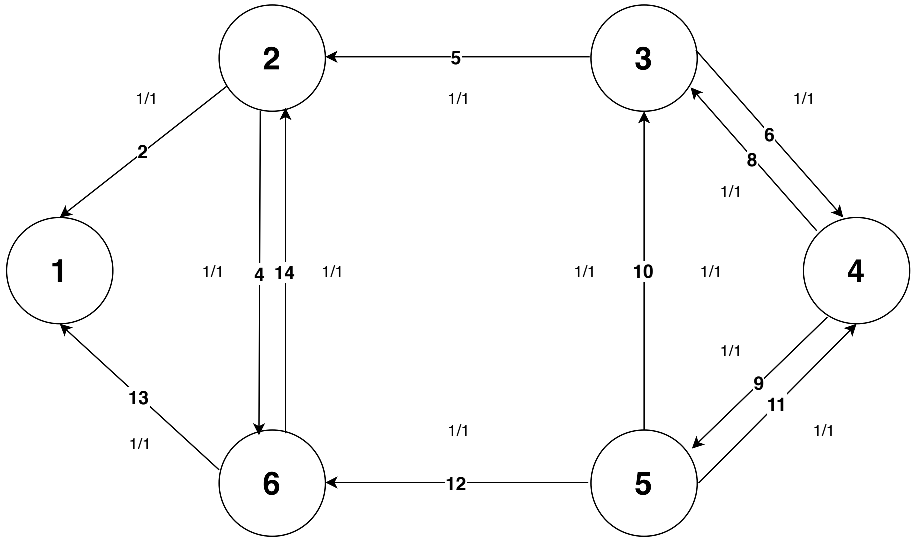

Figura 2.4: Final physical topology graph.

opticalMultiplexing-SystemIndex	sourceNode	destination-Node	numberOf-Wavelenghts	wavelenghts (nm)	available-Wavelenghts
0	1	2	2	[1310 1550]	[0 0]
1	1	6	2	[1310 1550]	[0 0]
2	2	1	2	[1310 1550]	[1 1]
3	2	3	2	[1310 1550]	[0 0]
4	2	6	2	[1310 1550]	[1 1]
5	3	2	2	[1310 1550]	[1 1]
6	3	4	2	[1310 1550]	[0 1]
7	3	5	2	[1310 1550]	[0 0]
8	4	3	2	[1310 1550]	[1 1]
9	4	5	2	[1310 1550]	[0 1]
10	5	3	2	[1310 1550]	[1 1]
11	5	4	2	[1310 1550]	[1 1]
12	5	6	2	[1310 1550]	[1 1]
13	6	1	2	[1310 1550]	[1 1]
14	6	2	2	[1310 1550]	[1 1]
15	6	5	2	[1310 1550]	[0 0]

Tabela 2.112: Final opticalMultiplexingSystems variable values.

2.7.12 Transparent with 1+1 Protection

Student Name	:	Pedro Coelho (01/03/2018 -)
Goal	:	Implement the heuristic model for the transparent transport mode with 1 plus 1 protection.

Contrary to the transparent without survivability transport mode, the transparent with 1+1 protection technique has a backup path, so if there is a network failure it is more likely to not suffer large data losses. The backup paths are always different from the primary ones and they prevent that the information going through optical channels could be lost in these occasions. However, the CAPEX will be significantly higher (more than the double), because that includes a secondary path that will increase several network elements.

After the creation of the matrices and the network topology, it is necessary to apply the routing and grooming algorithms created. In the end, a report algorithm will be applied to obtain the best CAPEX result for the network in question.

We also must take into account the following particularity of this mode of transport:

- $N_{OXC,n}$ = 1, $\forall$ n that process traffic
- $N_{EXC,n}$ = 1, $\forall$ n that process traffic

The minimization of the network CAPEX is made through the equation 1.1 where in this case for the cost of nodes we have in consideration the electric cost 1.4 and the optical cost 1.5.

In this case the value of $P_{exc,c,n}$ is obtained by equation 2.1 for short-reach and by the equation 2.2 for long-reach and the value of $P_{oxc,n}$ is obtained by equation 2.3.

The equation 2.1 refers to the number of short-reach ports of the electrical switch with bit-rate c in node n, $P_{exc,c,n}$, i.e. the number of tributary ports with bit-rate c in node n which can be calculated as

$$P_{exc,c,n} = \sum_{d=1}^{N} D_{nd,c} \tag{2.1}$$

where $D_{nd,c}$ are the client demands between nodes n and d with bit rate c.

In this case there is the following particularity:

- When n=d the value of client demands is always zero, i.e, $D_{nn,c} = 0$

As previously mentioned, the equation 2.2 refers to the number of long-reach ports of the electrical switch with bit-rate -1 in node n, $P_{exc,-1,n}$, i.e. the number of add ports of node n which can be calculated as

$$P_{exc,-1,n} = \sum_{j=1}^{N} f_{nj}^{od} \tag{2.2}$$

where f_{nj}^{od} is the number of optical channels between node n and node j for all demand pairs (od).

The equation 2.3 refers to the number of line ports and the number of adding ports of node n which can be calculated as

$$P_{oxc,n} = \sum_{j=1}^{N} 2f_{nj}^{od} + \sum_{j=1}^{N} \lambda_{nj} \tag{2.3}$$

where f_{nj}^{od} refers to the number of line ports for all demand pairs (od) and λ_{nj} refers to the number of add ports.

To implement this heuristic approach there are used algorithms made in Java in a programming software called Eclipse and they are tested in an open-source network program called Net2Plan. In the Net2Plan guide section **??** there is an explanation on how to use and test them in this network planner.

In the next pages it will be described all the steps performed to obtain the final results in the transparent transport mode with 1+1 protection. In the figure below 2.5 it is shown a fluxogram with the description of this transport mode approach.

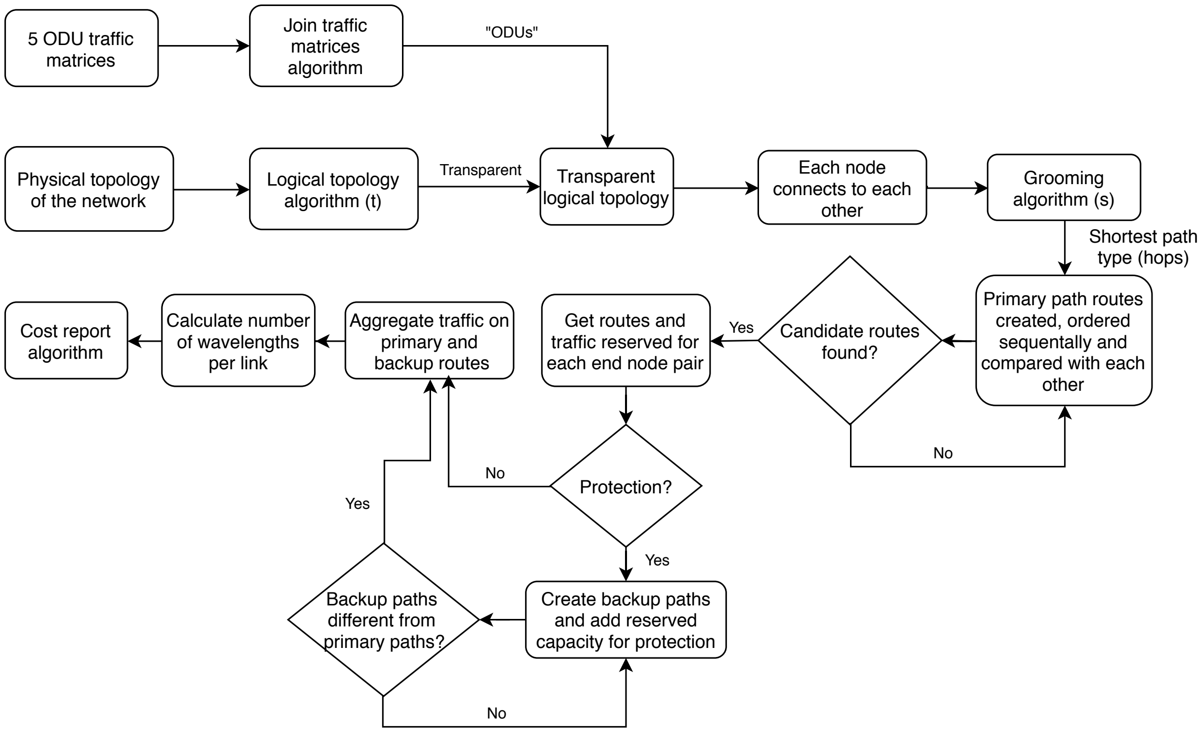

Figura 2.5: Fluxogram with the steps performed in the transparent with 1+1 protection transport mode approach.

Creation and join the traffic matrices

The first step is to create the traffic matrices based on the reference network **??**. In order to create the 5 traffic matrices in Net2Plan it is necessary the length of all the links and the total traffic used in this network, so later it is needed to define in Net2Plan the length in all end nodes and the total traffic depends on the value of traffic used (low traffic - 0.5 Tbit/s, medium traffic - 5 Tbit/s and high traffic - 10 Tbit/s). As you can see in the figure below, it is defined the path of the 5 ODUs and they will be aggregated in just one single ODU, making it possible to join all the demands in just one file and load it later into the network. This final resulting ODU joins the multiple traffic demands from all the traffic matrices previously created and, of course, the traffic demands will depend on the values used on the creation of the matrices (low, medium and high traffic).

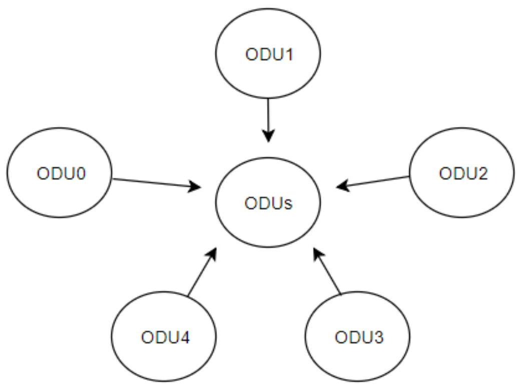

Figura 2.6: Join the 5 ODU traffic matrices into 1 single file "ODUs". The 5 traffic demands from the traffic matrices previously created are joined into 1 file to load it later on Net2Plan.

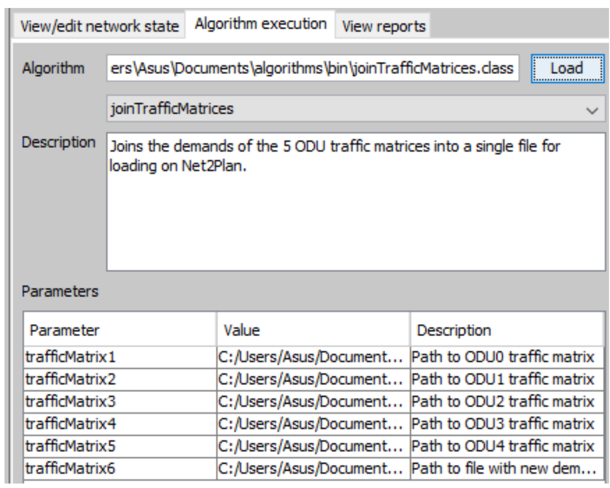

Figura 2.7: Load of the join traffic matrices algorithm for the transparent transport mode on Net2Plan. It is defined the 5 paths to load the 5 ODU traffic matrices and the last path is the one where will be saved the file that joins all 5 the traffic demands.

Creation of the physical topology

The next step is to create the allowed physical topology of the network in Net2Plan. This network consists in 6 nodes and 8 bidirectional links. It is now also possible to define the length in all links. In the figure below it is shown the allowed physical topology in this transport mode.

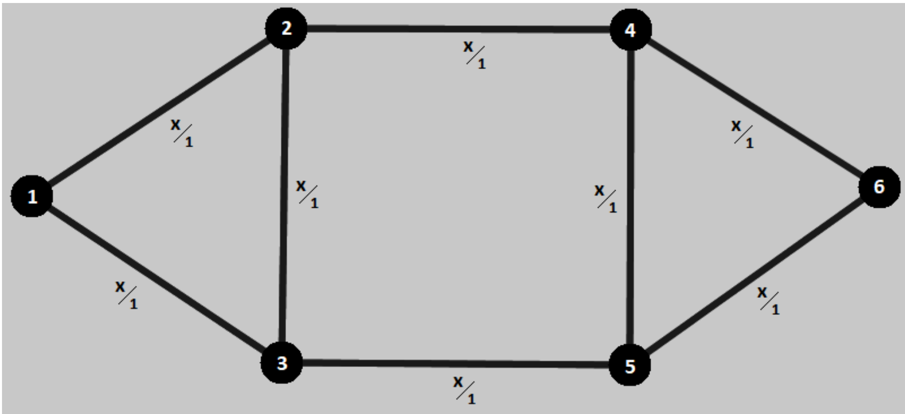

Figura 2.8: Allowed physical topology. The allowed physical topology is defined by the duct and sites in the field. It is assumed that each duct supports up to 1 bidirectional transmission system and each site supports up to 1 node.

Creation of the logical topology

It is now time to create the allowed logical topology. A network topology represents how the links and the nodes of the network interconnect with each other and the logical topology algorithm creates the logical topology on another layer. In the transparent transport mode each node connects to each other creating direct links between all nodes in the network. Going through all nodes, if a node has a different index from other one, then creates a shortest and direct link between them. These additions of links between end nodes are made in the new upper layer of the network. The respective demands are saved in the new upper layer and those demands from the lower layer are then removed. The lower layer is the physical layer of the network and it is now created a new upper layer which is the logical layer of the network and represents the logical topology of the transparent transport mode. The allowed physical and optical topologies, the logical topologies for all ODUs and the resulting physical topology is shown in the next section below 2.7.12 for the three traffic scenarios. It is shown below three figures with the code in Java of the creation of the network logical topology, the load of the logical topology algorithm in Net2Plan and the resulting allowed optical topology for the transparent transport mode with 1+1 protection.

```java
if(netPlan.isSingleLayer() && logicalTopology.equalsIgnoreCase("Transparent"))
{
    this.lowerLayer = netPlan.getNetworkLayerDefault();
    lowerLayer.setName("Physical Topology");
    this.upperLayer = netPlan.addLayer("Logical Topology Transparent","Upper layer of the design","ODU","ODU",null);
    upperLayer.setDescription("Transparent Logical Topology");
    netPlan.removeAllLinks(upperLayer);

    for (Node i : netPlan.getNodes()) {
        for (Node j : netPlan.getNodes()) {
            if (i.getIndex() != j.getIndex())
            {
                netPlan.addLink(i, j, 0 , netPlan.getNodePairEuclideanDistance(i, j), 200000 , null , upperLayer);
            }
        }
    }
}
```

Figura 2.9: Java code of the logical topology approach for the transparent transport mode. The logical layer is created by adding direct links between all end nodes. The new layer is now the transparent logical topology of the network.

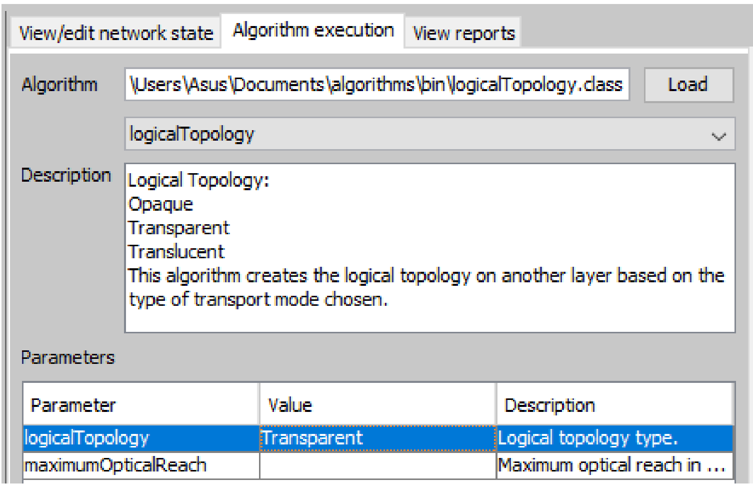

Figura 2.10: Load of the logical topology algorithm for the transparent transport mode.

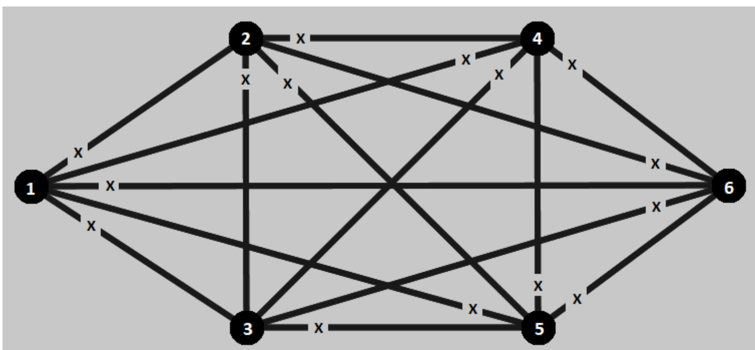

Figura 2.11: Allowed optical topology. It is assumed that each connections between demands supports up to 100 lightpaths.

Creation of routes and aggregation of traffic

After a network topology is created, it is now time to set the routing algorithm. In the transparent with 1+1 protection transport mode the routing algorithm is similar with the one used in opaque transport mode. It starts with going through all the demands and nodes which have different index between them (end nodes), create bidirectional routes (in this case the primary paths) based on the shortest path Dijkstra algorithm and then search the candidate routes for the respective demand. In this report it is used the shortest path type in hops. These routes are ordered sequentially and the shortest one per each demand is the primary path. The demands from the lower layer are removed and then saved in the upper layer. As we also have a dedicated 1+1 protection scheme, if the network has this feature active, the algorithm will compare the previous candidate routes that will be saved to a list with the new ones that will be created. If the new routes are different from the previously created ones and if they are the next shortest path routes, then the algorithm will add these routes to the network and they will be the protection segments (backup paths) of the network. The offered traffic demands will be also set into these protection path routes. The routes are saved to a "Set" of routes and in each link of end nodes it is set the traffic demands into these routes that will integrate the whole network. The final resulting backup path routes are used to prevent network failures. Despite of the fact the network will be much more secure, the network CAPEX will increase more than the double, due to the creation of primary and backup paths.

```
case "Logical Topology Transparent":

    for (Demand d : netPlan.getDemands(lowerLayer)) {
        boolean odd=true;
        int counter=0;

        Set<Route> droutes = d.getRoutes();
        System.out.println("droutes: " + droutes.size());

        for(Route c: droutes) {
            counter++;
            boolean jump=false;

            if(odd) {
                c.setCarriedTraffic(d.getOfferedTraffic(), d.getOfferedTraffic());
                save=c;
                odd=false;
                System.out.println("Roots");
            }
```

Figura 2.12: Creation of routes and aggregation of traffic for the transparent with 1+1 protection transport mode. The candidate routes are searched by the shortest path type and the offered traffic demands are set into these routes.

```
        else {
            if (protection) {
                List<Link>  workingpath = save.getSeqLinksRealPath();
                System.out.println("Protection-Transparent");
                for(Link t:workingpath) {
                    if(c.getSeqLinksRealPath().contains(t)) {
                        jump=true;
                        break;
                    }
                }

                if(jump==false) {
                    ProtectionSegment segment=netPlan.addProtectionSegment(c.getSeqLinksRealPath(),
                                                                  d.getOfferedTraffic(), null);
                    save.addProtectionSegment(segment);
                    odd=true;
                    break;
                }

                if(jump==true && counter == droutes.size()) {
                    ProtectionSegment segment=netPlan.addProtectionSegment(c.getSeqLinksRealPath(),
                                                                  d.getOfferedTraffic(), null);
                    save.addProtectionSegment(segment);
                    odd=true;
                    throw new Net2PlanException ("Number of routes is not enough");
                }
            }
        }
    }
}
```

Figura 2.13: Creation of routes and aggregation of traffic for the transparent with 1+1 protection transport mode. The protection segments are added to all the primary paths that were chosen by the shortest path type method.

```
ArrayList<Long> tNodeIds = netPlan.getNodeIds();
Node in;
Node out;
Set<Route> groomRoute;
Set<ProtectionSegment> protectRoutes;
Route compare=null;
ProtectionSegment compare1=null;
List<Link> path;
int nW=0;

for (long tNodeId : tNodeIds)   {
    in = netPlan.getNodeFromId(tNodeId);
    for (long tNodeId1 : tNodeIds)  {

        if(tNodeId==tNodeId1)continue;

        out = netPlan.getNodeFromId(tNodeId1);
        double totaltraffic=0;

        groomRoute=netPlan.getNodePairRoutes(in,out,false,lowerLayer);
        protectRoutes=netPlan.getNodePairProtectionSegments(in,out,false,lowerLayer);

        for(Route d:groomRoute)
        {
            totaltraffic = totaltraffic + d.getCarriedTraffic();
            compare=d;
        }

        path=compare.getSeqLinksRealPath();
```

Figura 2.14: Creation of routes and aggregation of traffic for the transparent with 1+1 protection transport mode. The traffic demands are set in the candidate primary path routes found earlier and also compared with the backup path routes. The traffic demands are also set into these protection path routes.

```
//Protection Segments
totaltraffic=0;

for(ProtectionSegment protect:protectRoutes) {
    totaltraffic = totaltraffic + protect.getReservedCapacityForProtection();
    compare1 = protect;
}

if (protection) {
    path=compare1.getSeqLinks();
}
```

Figura 2.15: Creation of routes and aggregation of traffic for the transparent with 1+1 protection transport mode. The traffic demands are set in the candidate primary path routes found earlier and also compared with the backup path routes. The traffic demands are also set in these protection path routes.

Function	Definition
netPlan.getDemands(lowerLayer)	Returns the array of demands for the lower layer.
d.getRoutes()	Returns all the routes associated to the demand "d".
c.setCarriedTraffic()	Sets the route carried traffic and the occupied capacity in the links, setting it up to be the same in all links.
d.getOfferedTraffic()	Returns the offered traffic of the demand "d".
save.getSeqLinksRealPath()	Returns the links of routes ordered sequentially.
save.addProtectionSegment(segment)	Add "segment" as a protection path in the route "save".
netPlan.getNodeIds()	Returns the array of the nodes' indexes.
netPlan.getNodeFromId(tNodeId)	Returns the node with the index "tNodeId".
netPlan.getNodePairRoutes (in,out,false,lowerLayer)	Returns the routes at "lowerLayer" from nodes "in" and "out".
netPlan.getNodePairProtection Segments(in,out,false,lowerLayer)	Returns the protection segments at "lowerLayer" from nodes "in" and "out".
protect.getReservedCapacity ForProtection()	Returns the link capacity reserved for protection segments.

Tabela 2.113: Table with the description of the main functions in the creation of routes and aggregation of traffic in the grooming algorithm.

Calculation of the number of wavelengths per link

The final step of the routing and grooming algorithms is to calculate the number of wavelengths per link for the whole network. This is the last and an important step because with the number of wavelengths per link in the network, it is possible to calculate other network components. In the transparent transport mode, as in the figure below shows, the algorithm starts with going through all the nodes which have different index between them (end nodes) and in all the links that crosses between these pairs of nodes is reserved a link capacity based on the previous traffic aggregation. The total carried traffic in the link including protection and non-protection segments will be divided by the wavelength capacity and it is now possible to obtain the number of wavelengths per link.

```
for (Link link:path)
{
    String nw = link.getAttribute("nW");
    nW=0;

    if(nw!=null)
    {
        nW = Integer.parseInt(nw);
        nW = (int) (nW+Math.ceil(totaltraffic/wavelengthCapacity));
        link.setAttribute("nW",String.valueOf(nW));
    }else {
        nW = (int) Math.ceil(totaltraffic/wavelengthCapacity);
        link.setAttribute("nW",String.valueOf(nW));
    }
}
```

Figura 2.16: Calculation of the number of wavelengths per link for the transparent transport mode. The link capacity is reserved based on the previous traffic aggregation.

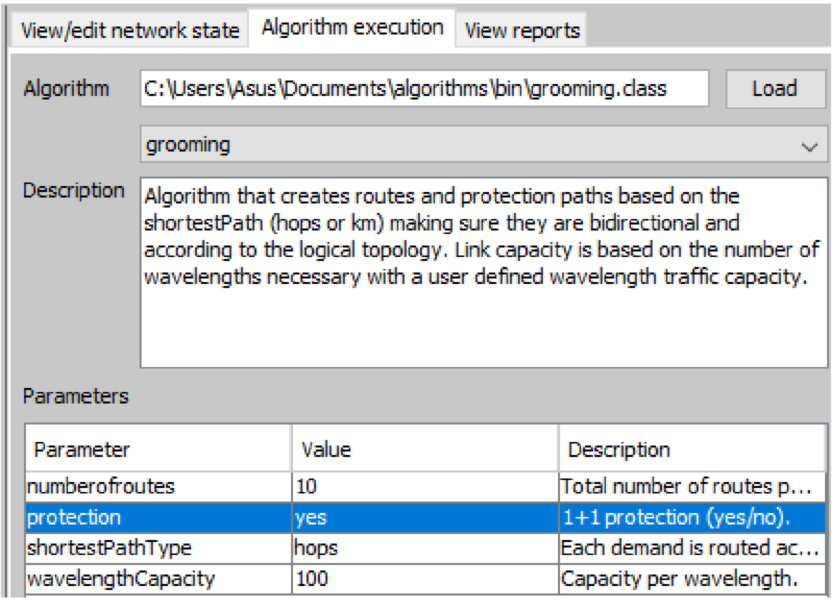

Figura 2.17: Load of the grooming algorithm for the transparent with 1+1 protection transport mode. The total number of routes per demand is set to 10, the user can define if the model is with or without protection, the shortest path type is set to "hops" and the capacity per wavelength is used 100 optical channels.

Network cost report

In order to obtain the network CAPEX results, the formulas needed to calculate the network elements and that are demonstrated previously in the beginning of this section 2.7.12 were "translated" into Java code in a cost report algorithm. This algorithm can be loaded in Net2Plan and calculates and shows in tables the network CAPEX and also the per-link and per-node information with more details.

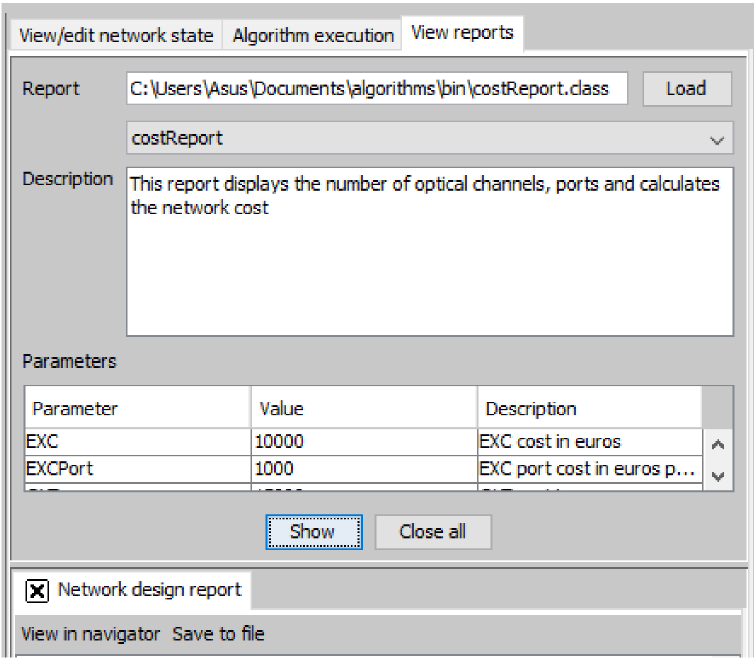

Figura 2.18: Load of the cost report algorithm on Net2Plan. The result view is an HTML page with the network optical and electrical components and their costs.

Result description

It is already known all the necessary formulas to obtain the CAPEX value for the reference network **??**. As described in the subsection of the network traffic **??**, it is necessary to obtain three different values of CAPEX for the low (0.5 Tbit/s), medium (5 Tbit/s) and high (10 Tbit/s) traffic. It is used a network software program called Net2Plan which can design the traffic matrices, create all the network topologies, simulate the algorithms into the network implemented in the programming software called Eclipse and analyze the results obtained. In this chapter will be demonstrated the results by Vasco's heuristics from 2016. In each of the three traffic scenarios, it will be shown the network topologies followed by the table with the CAPEX value of the network.

Low Traffic Scenario:

In this scenario we have to take into account the traffic calculated in **??**. In a first phase we will show the various existing topologies of the network. The first are the allowed topologies, physical and optical topologies, the second are the logical topology for all ODUs and finally the resulting physical topology.

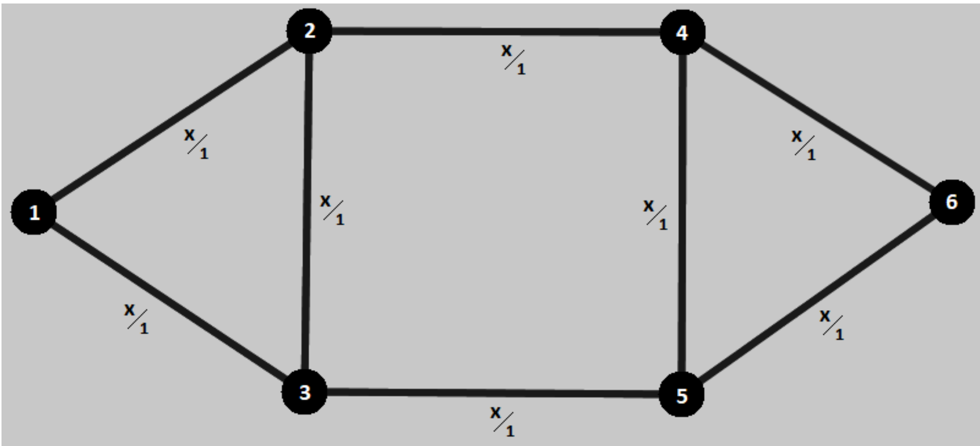

Figura 2.19: Allowed physical topology. The allowed physical topology is defined by the duct and sites in the field. It is assumed that each duct supports up to 1 bidirectional transmission system and each site supports up to 1 node.

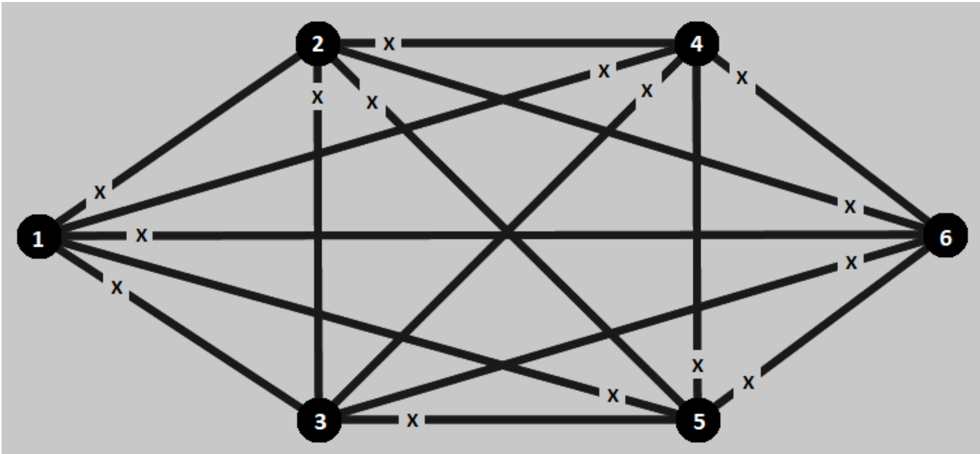

Figura 2.20: Allowed optical topology. The allowed optical topology is defined by the transport mode (transparent transport mode in this case). It is assumed that each connections between demands supports up to 100 lightpaths.

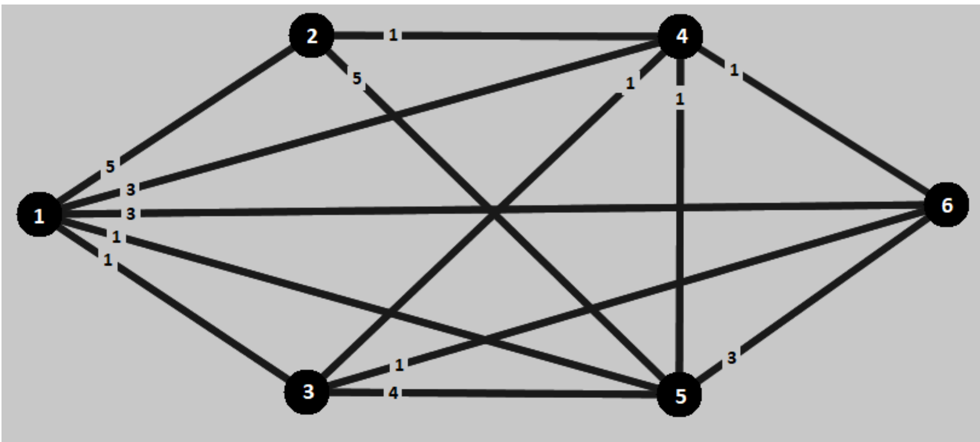

Figura 2.21: ODU0 logical topology defined by the ODU0 traffic matrix.

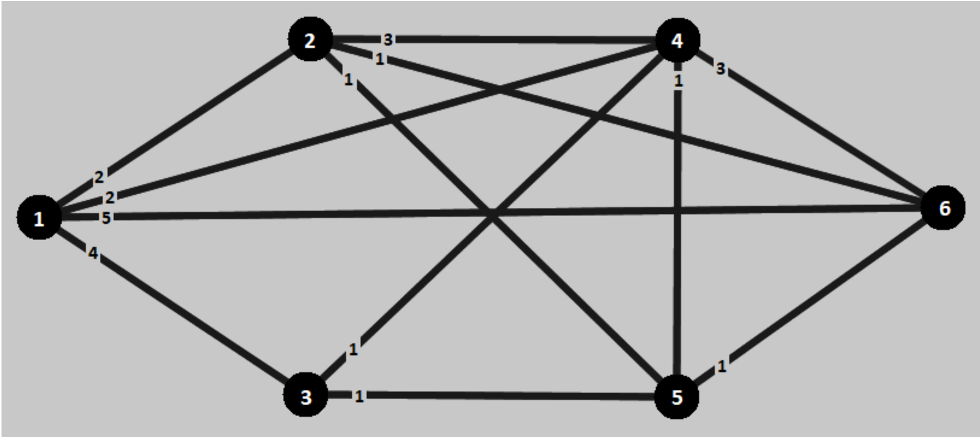

Figura 2.22: ODU1 logical topology defined by the ODU1 traffic matrix.

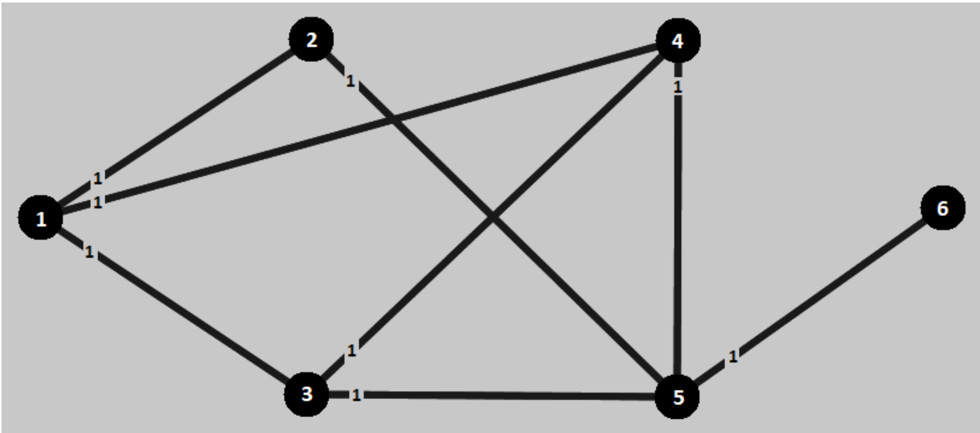

Figura 2.23: ODU2 logical topology defined by the ODU2 traffic matrix.

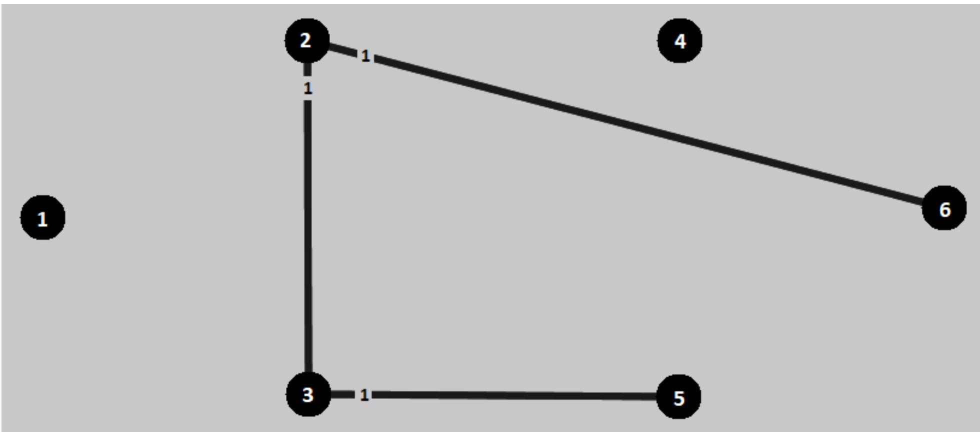

Figura 2.24: ODU3 logical topology defined by the ODU3 traffic matrix.

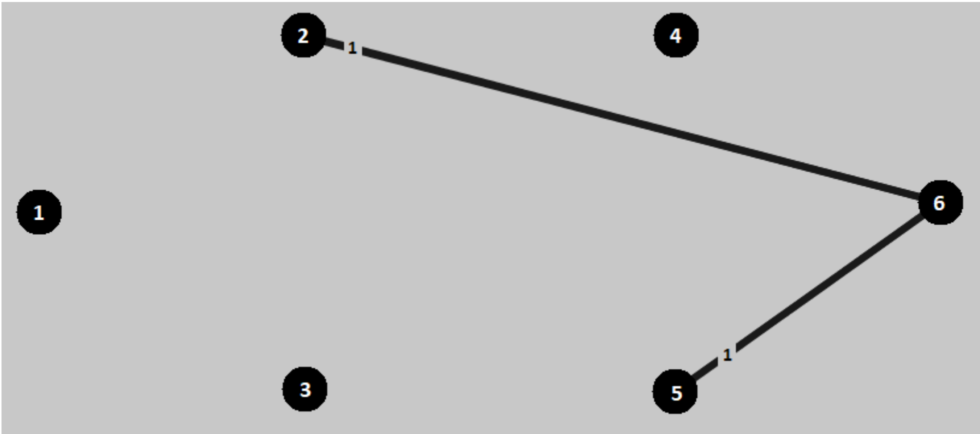

Figura 2.25: ODU4 logical topology defined by the ODU4 traffic matrix.

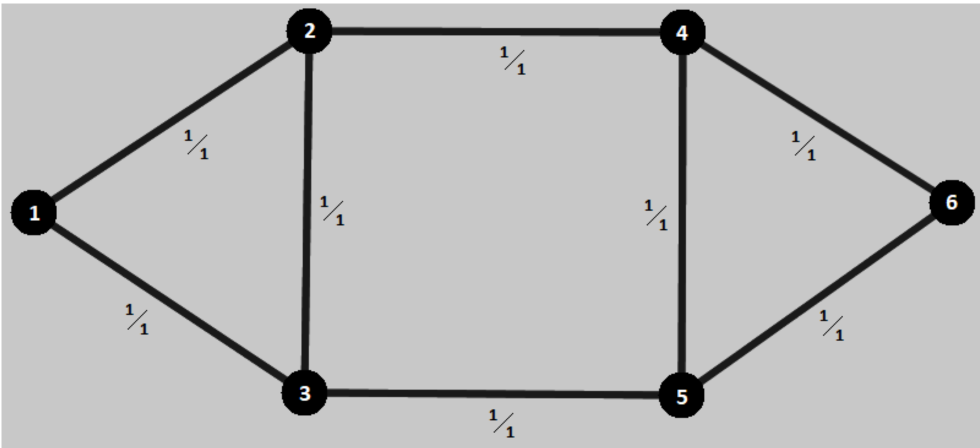

Figura 2.26: Physical topology after dimensioning.

Following all the steps mentioned in the **??**, applying the routing and grooming heuristic algorithms in the Net2Plan software and using all the data referring to this scenario, the obtained result for the Vasco's heuristics can be consulted in the following table 2.114. In table 1.65 mentioned in previous model we can see how all the values were calculated.

CAPEX of the Network						
			Quantity	Unit Price	Cost	Total
Link Cost	OLTs		16	15 000 €	240 000 €	68 520 000 €
	100 Gbits/s Transceivers		136	5 000 €/Gbit/s	68 000 000 €	
	Amplifiers		70	4 000 €	280 000 €	
Node Cost	Electrical	EXCs	6	10 000 €	60 000 €	4 007 590 €
		ODU0 Ports	60	10 €/port	600 €	
		ODU1 Ports	50	15 €/port	750 €	
		ODU2 Ports	16	30 €/port	480 €	
		ODU3 Ports	6	60 €/port	360 €	
		ODU4 Ports	4	100 €/port	400 €	
		Transponders	34	100 000 €/port	3 400 000 €	
	Optical	OXCs	6	20 000 €	120 000 €	
		Line Ports	136	2 500 €/port	340 000 €	
		Add Ports	34	2 500 €/port	85 000 €	
Total Network Cost						72 527 590 €

Tabela 2.114: Table with detailed description of CAPEX of Vasco's 2016 results.

Medium Traffic Scenario:

In this scenario we have to take into account the traffic calculated in **??**. In a first phase we will show the various existing topologies of the network. The first are the allowed topologies, physical and optical topologies, the second are the logical topology for all ODUs and finally the resulting physical topology.

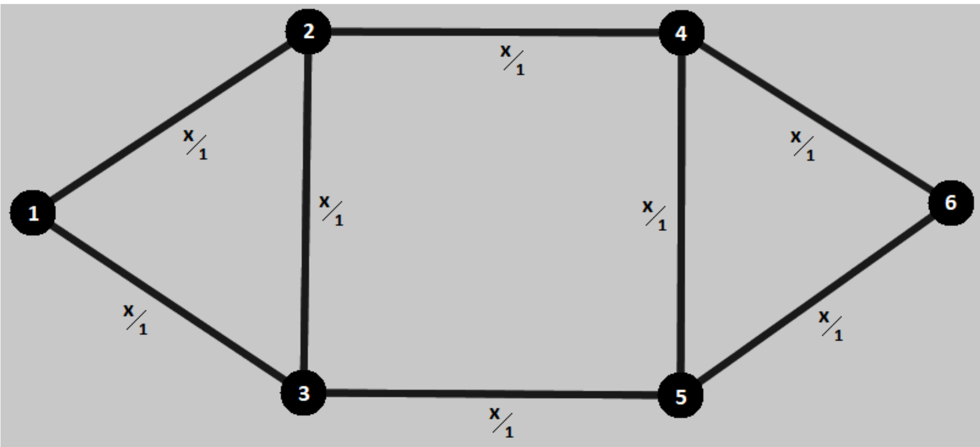

Figura 2.27: Allowed physical topology. The allowed physical topology is defined by the duct and sites in the field. It is assumed that each duct supports up to 1 bidirectional transmission system and each site supports up to 1 node.

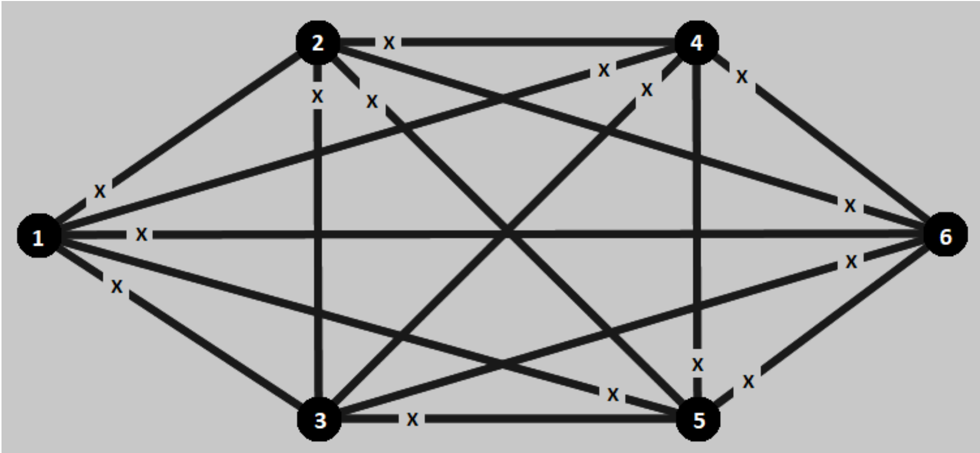

Figura 2.28: Allowed optical topology. The allowed optical topology is defined by the transport mode (transparent transport mode in this case). It is assumed that each connections between demands supports up to 100 lightpaths.

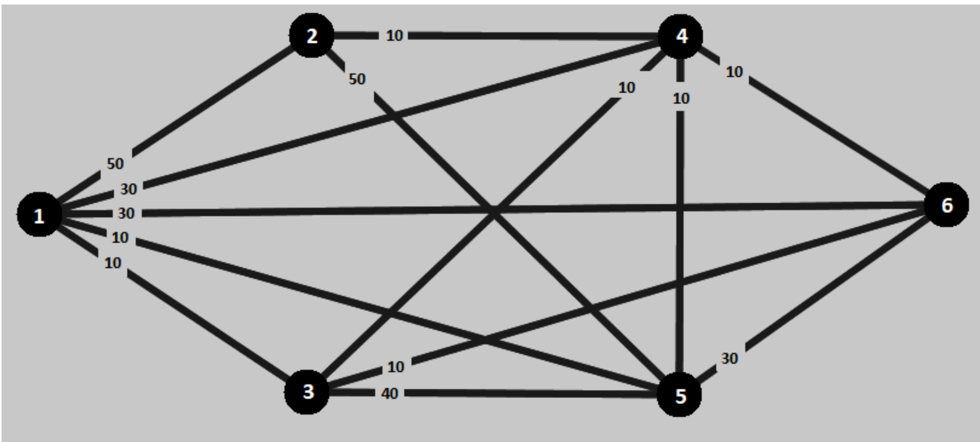

Figura 2.29: ODU0 logical topology defined by the ODU0 traffic matrix.

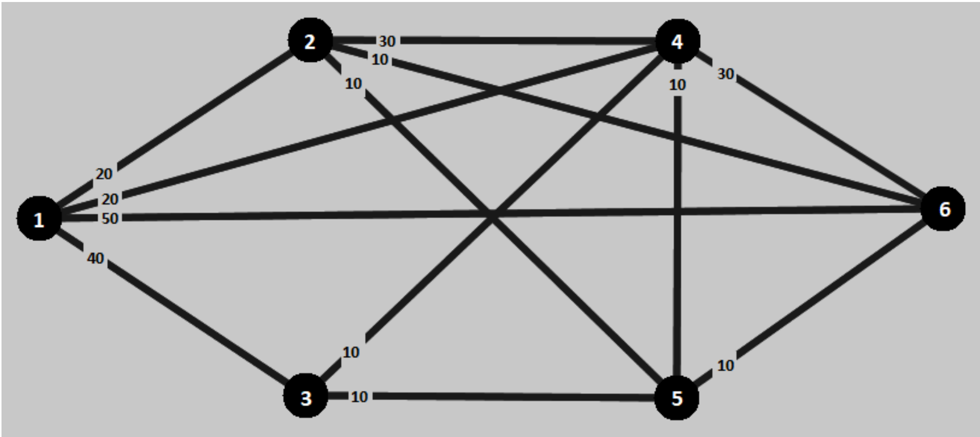

Figura 2.30: ODU1 logical topology defined by the ODU1 traffic matrix.

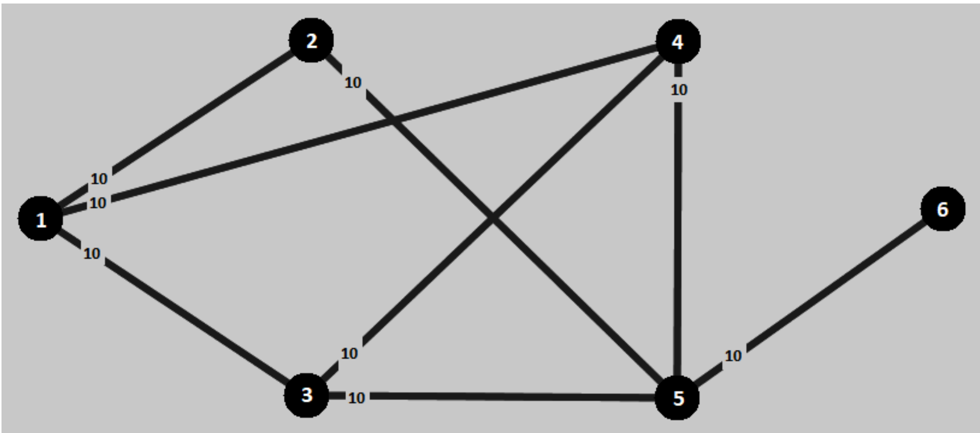

Figura 2.31: ODU2 logical topology defined by the ODU2 traffic matrix.

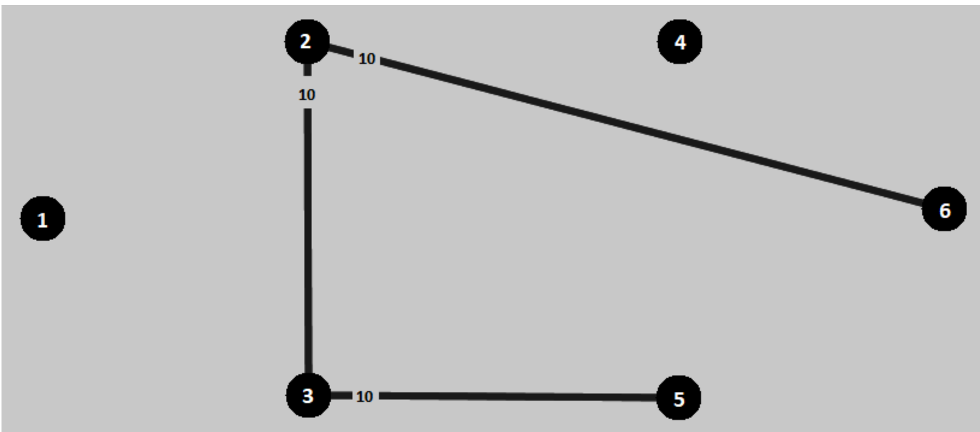

Figura 2.32: ODU3 logical topology defined by the ODU3 traffic matrix.

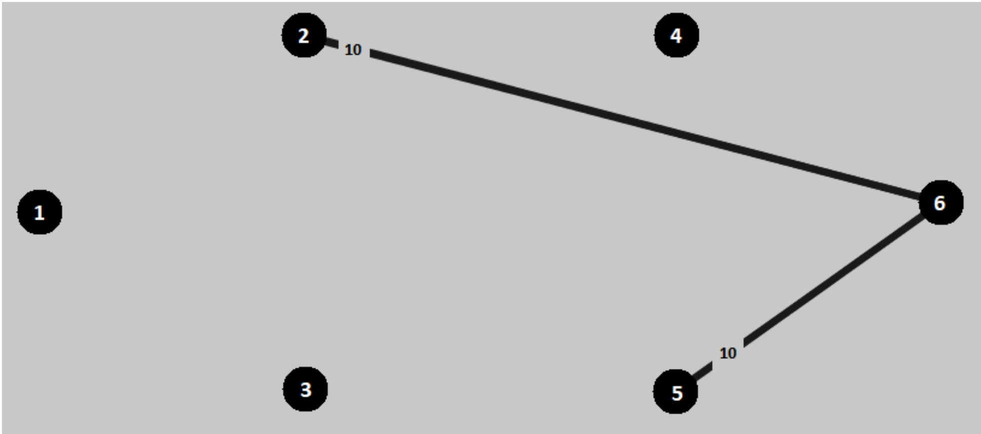

Figura 2.33: ODU4 logical topology defined by the ODU4 traffic matrix.

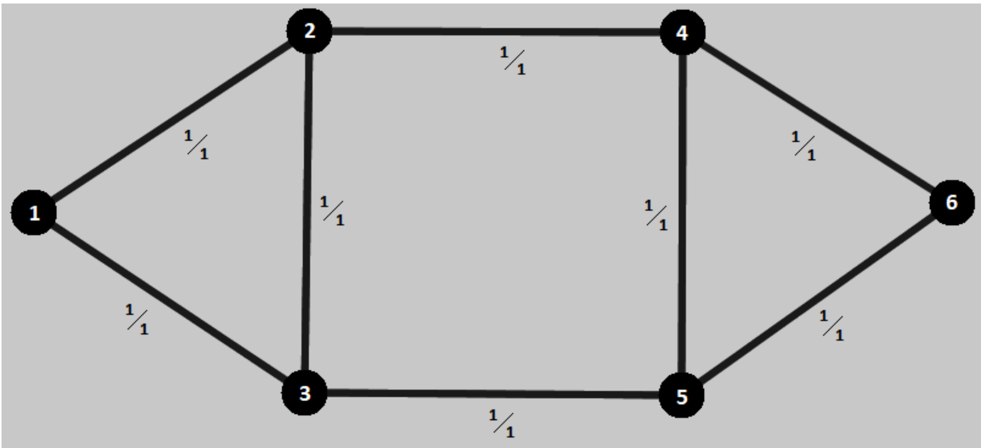

Figura 2.34: Physical topology after dimensioning.

Following all the steps mentioned in the **??**, applying the routing and grooming heuristic algorithms in the Net2Plan software and using all the data referring to this scenario, the obtained result for the Vasco's heuristics can be consulted in the following table 2.115. In table 1.65 mentioned in previous model we can see how all the values were calculated.

CAPEX of the Network						
			Quantity	Unit Price	Cost	Total
Link Cost	OLTs		16	15 000 €	240 000 €	226 520 000 €
	100 Gbits/s Transceivers		452	5 000 €/Gbit/s	226 000 000 €	
	Amplifiers		70	4 000 €	280 000 €	
Node Cost	Electrical	EXCs	6	10 000 €	60 000 €	15 890 900 €
		ODU0 Ports	600	10 €/port	6 000 €	
		ODU1 Ports	500	15 €/port	7 500 €	
		ODU2 Ports	160	30 €/port	4 800 €	
		ODU3 Ports	60	60 €/port	3 600 €	
		ODU4 Ports	40	100 €/port	4 000 €	
		Transponders	142	100 000 €/port	14 200 000 €	
	Optical	OXCs	6	20 000 €	120 000 €	
		Line Ports	452	2 500 €/port	1 130 000 €	
		Add Ports	142	2 500 €/port	355 000 €	
Total Network Cost						242 410 900 €

Tabela 2.115: Table with detailed description of CAPEX of Vasco's 2016 results.

High Traffic Scenario:

In this scenario we have to take into account the traffic calculated in **??**. In a first phase we will show the various existing topologies of the network. The first are the allowed topologies, physical and optical topologies, the second are the logical topology for all ODUs and finally the resulting physical topology.

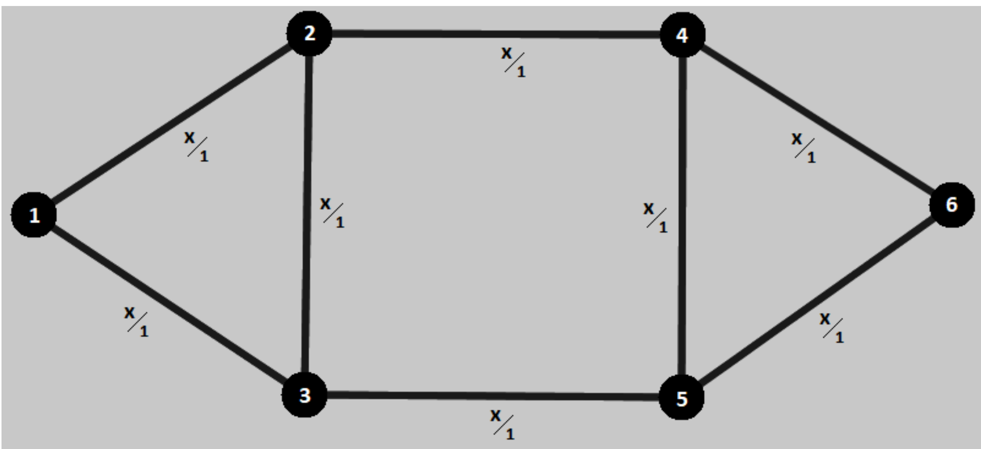

Figura 2.35: Allowed physical topology. The allowed physical topology is defined by the duct and sites in the field. It is assumed that each duct supports up to 1 bidirectional transmission system and each site supports up to 1 node.

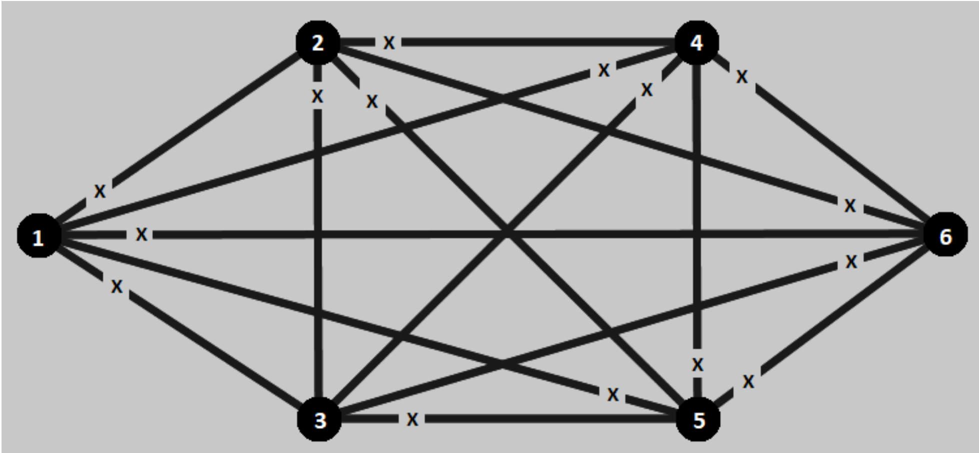

Figura 2.36: Allowed optical topology. The allowed optical topology is defined by the transport mode (transparent transport mode in this case). It is assumed that each connections between demands supports up to 100 lightpaths.

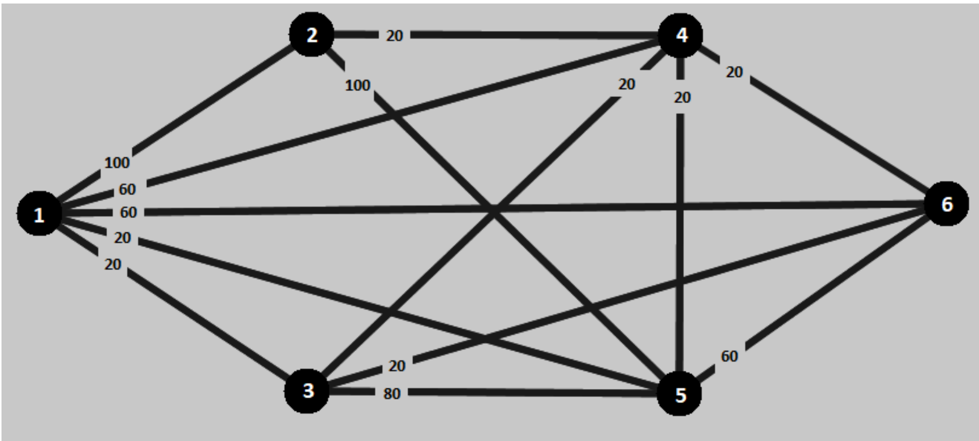

Figura 2.37: ODU0 logical topology defined by the ODU0 traffic matrix.

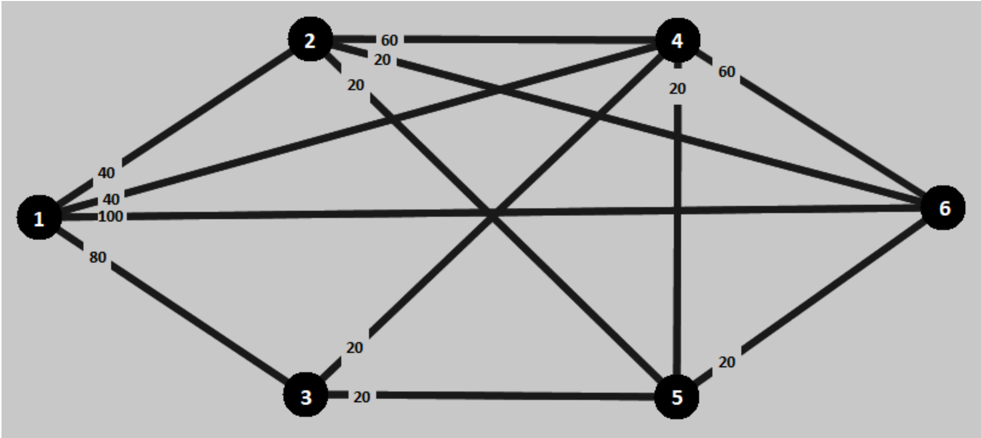

Figura 2.38: ODU1 logical topology defined by the ODU1 traffic matrix.

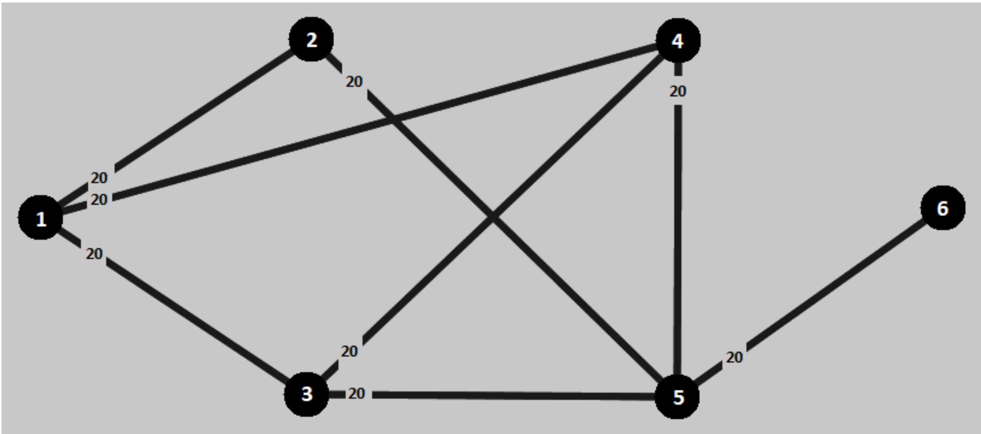

Figura 2.39: ODU2 logical topology defined by the ODU2 traffic matrix.

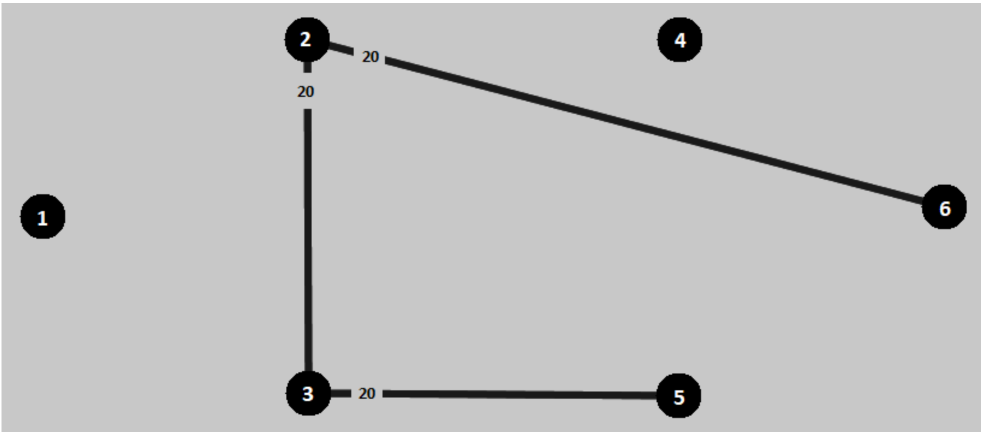

Figura 2.40: ODU3 logical topology defined by the ODU3 traffic matrix.

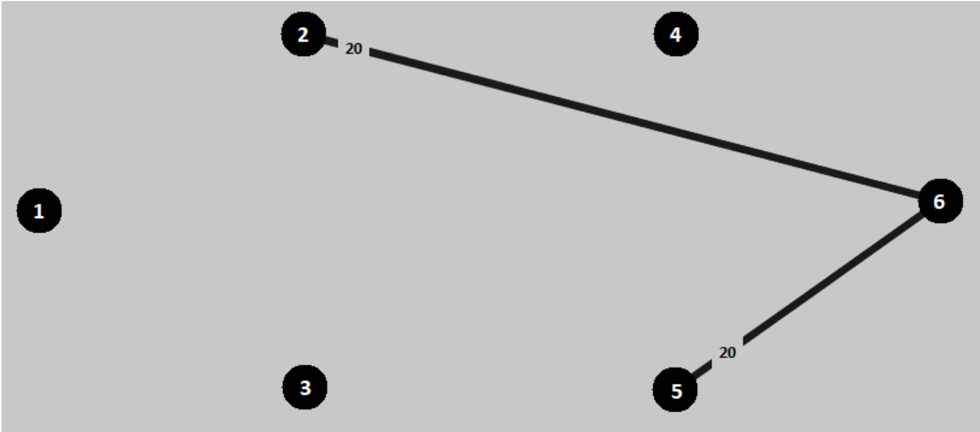

Figura 2.41: ODU4 logical topology defined by the ODU4 traffic matrix.

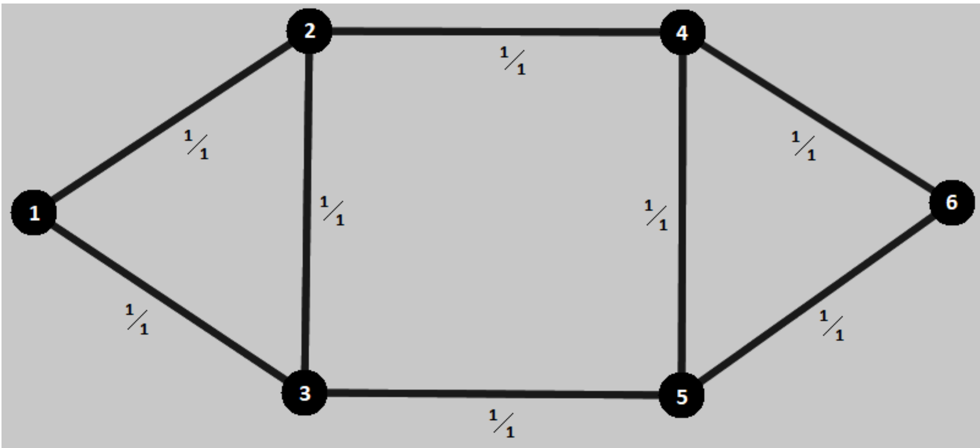

Figura 2.42: Physical topology after dimensioning.

Following all the steps mentioned in the **??**, applying the routing and grooming heuristic algorithms in the Net2Plan software and using all the data referring to this scenario, the obtained result for the Vasco's heuristics can be consulted in the following table 2.116. In table 1.65 mentioned in previous model we can see how all the values were calculated.

CAPEX of the Network						
			Quantity	Unit Price	Cost	Total
Link Cost	OLTs		16	15 000 €	240 000 €	424 520 000 €
	100 Gbits/s Transceivers		848	5 000 €/Gbit/s	424 000 000 €	
	Amplifiers		70	4 000 €	280 000 €	
Node Cost	Electrical	EXCs	6	10 000 €	60 000 €	29 821 800 €
		ODU0 Ports	1 200	10 €/port	12 000 €	
		ODU1 Ports	1 000	15 €/port	15 000 €	
		ODU2 Ports	320	30 €/port	9 600 €	
		ODU3 Ports	120	60 €/port	7 200 €	
		ODU4 Ports	80	100 €/port	8 000 €	
		Transponders	268	100 000 €/port	26 800 000 €	
	Optical	OXCs	6	20 000 €	120 000 €	
		Line Ports	848	2 500 €/port	2 120 000 €	
		Add Ports	268	2 500 €/port	670 000 €	
Total Network Cost						454 341 800 €

Tabela 2.116: Table with detailed description of CAPEX of Vasco's 2016 results.

Conclusions

Once we have obtained the results for all scenarios for the transparent without survivability and transparent with 1+1 protection we will now draw some conclusions about these results. For a better analysis of the results will be created the table **??** with the number of line ports, tributary ports and transceivers because they are important values for the cost of CAPEX, the cost of links, the cost of nodes and finally the cost of CAPEX.

	Low Traffic	Medium Traffic	High Traffic
CAPEX without survivability	30 317 590 €	99 700 900 €	186 006 800 €
CAPEX/Gbit/s without survivability	60 635 €/Gbit/s	19 940 €/Gbit/s	18 600 €/Gbit/s
Traffic (Gbit/s)	500	5 000	10 000
Bidirectional Links used	8	8	8
Number of Add ports	34	142	268
Number of Line ports	136	452	848
Number of Tributary ports	136	1 360	2 720
Number of Transceivers	136	452	848
Link Cost	68 520 000 €	226 520 000 €	454 520 000 €
Node Cost	4 007 590 €	15 890 900 €	29 821 800 €
CAPEX	**72 527 590 €**	**242 410 900 €**	**454 341 800 €**
CAPEX/Gbit/s	**145 055 €/Gbit/s**	**48 482 €/Gbit/s**	**45 434 €/Gbit/s**

Tabela 2.117: Table with different value of CAPEX for this case.

Looking at the previous table we can make some comparisons between the transparent with 1+1 protection scenario:

- Comparing the low traffic with the others we can see that despite having an increase of factor ten (medium traffic) and factor twenty (high traffic), the same increase does not occur in the final cost (it is lower);

  This happens because the number of the transceivers is lower than expected which leads by carrying the traffic with less network components and, consequently, the network CAPEX is lower;

- Comparing the medium traffic with the high traffic we can see that the increase of the factor is double and in the final cost this factor is very close but still inferior;

  This happens because the number of the transceivers is also lower but very close to the expected;

- Comparing the CAPEX cost per bit we can see that in the low traffic the cost is higher than the medium and high traffic, which in these two cases the value is similar, but still inferior in the higher traffic;

  This happens because the lower the traffic, the higher CAPEX/bit will be. We can see that in medium and high traffic the results tend to be one closer and lower value.

We can also make some comparisons between the transparent without survivability and transparent with 1+1 protection scenarios:

- We can see that in the transparent with 1+1 protection transport mode the CAPEX cost for all the three traffic is more than the double;

This happens because in the transparent with 1+1 protection transport mode there is a need of having a primary and a backup path, in case of a network failure, and the backup path is typically longer;

- Comparing the CAPEX cost per bit we can see that has a similar case in both of the two scenarios. In the low traffic the cost is higher than the medium and high traffic, which in these two cases the value is similar;

  This happens because the lower the traffic, the higher CAPEX/bit will be. We can see that in medium and high traffic the results tend to be one closer and lower value.

Opens Issues

The creation of this model for any scenario, started with some considerations and some open issues being:

- Allow blocking.

  The presented model assume that the solution is possible or impossible, does not support a partial solution where some demands are not routed (are blocked);

- Allow multiple transmission system.

  The presented model for each link only supports one transmission system.

2.7.13 Translucent without Survivability

Student Name	:	Pedro Coelho (01/03/2018 -)
Goal	:	Implement the heuristic model for the translucent transport mode without survivability.

The translucent networks (optical-bypass) consist of intermediate optical network architectures between opaque and transparent networks. In the translucent transport mode (multi-hop approach) an OEO conversion is performed when the optical signal falls below a pre-defined threshold. This threshold is called the maximum optical reach and it is the maximum distance that an optical signal can traverse in the optical domain. Then it is needed an wavelength regenerator in order to regenerate the optical signal and maintain the quality level in the transmission (QoT) of the signals. This allows the traversing of the optical signal in the optical domain as much as it can go through the path, from source to destination.

For cost savings, the translucent networks aims at using the minimum number of regenerators and wavelengths of the network, being the most advantageous transport mode in the optical backbone networks.

We also must take into account the following particularity of this mode of transport:

- $N_{OXC,n}$ = 1, $\forall$ n that process traffic
- $N_{EXC,n}$ = 1, $\forall$ n that process traffic

The minimization of the network CAPEX is made through the equation 1.1 where in this case for the cost of nodes we have in consideration the electric cost 1.4 and the optical cost 1.5.

In this case the value of $P_{exc,c,n}$ is obtained by equation 2.4 for short-reach and by the equation 2.5 for long-reach and the value of $P_{oxc,n}$ is obtained by equation 2.6.

The equation 2.4 refers to the number of short-reach ports of the electrical switch with bit-rate c in node n, $P_{exc,c,n}$, i.e. the number of tributary ports with bit-rate c in node n which can be calculated as

$$P_{exc,c,n} = \sum_{d=1}^{N} D_{nd,c} \tag{2.4}$$

where $D_{nd,c}$ are the client demands between nodes n and d with bit rate c.

In this case there is the following particularity:

- When n=d the value of client demands is always zero, i.e, $D_{nn,c} = 0$

As previously mentioned, the equation 2.5 refers to the number of long-reach ports of the electrical switch with bit-rate -1 in node n, $P_{exc,-1,n}$, i.e. the number of add ports of node n which can be calculated as

$$P_{exc,-1,n} = \sum_{j=1}^{N} \lambda_{nj} \tag{2.5}$$

where λ_{nj} is the number of optical channels between node n and node j.

The equation 2.6 refers to the number of ports in optical switch in node n, $P_{oxc,n}$, i.e. the number of line ports and the number of adding ports of node n which can be calculated as

$$P_{oxc,n} = \sum_{j=1}^{N} f_{nj}^{od} + \sum_{j=1}^{N} \lambda_{nj} \tag{2.6}$$

where f_{nj}^{od} refers to the number of line ports for all demand pairs (od) and λ_{nj} refers to the number of add ports.

To implement this heuristic approach there are used algorithms made in Java in a programming software called Eclipse and they are tested in an open-source network program called Net2Plan. In the next pages it will be described all the steps performed to obtain the final results in the translucent transport mode without survivability. In the figure below 2.43 it is shown a fluxogram with the description of this transport mode approach.

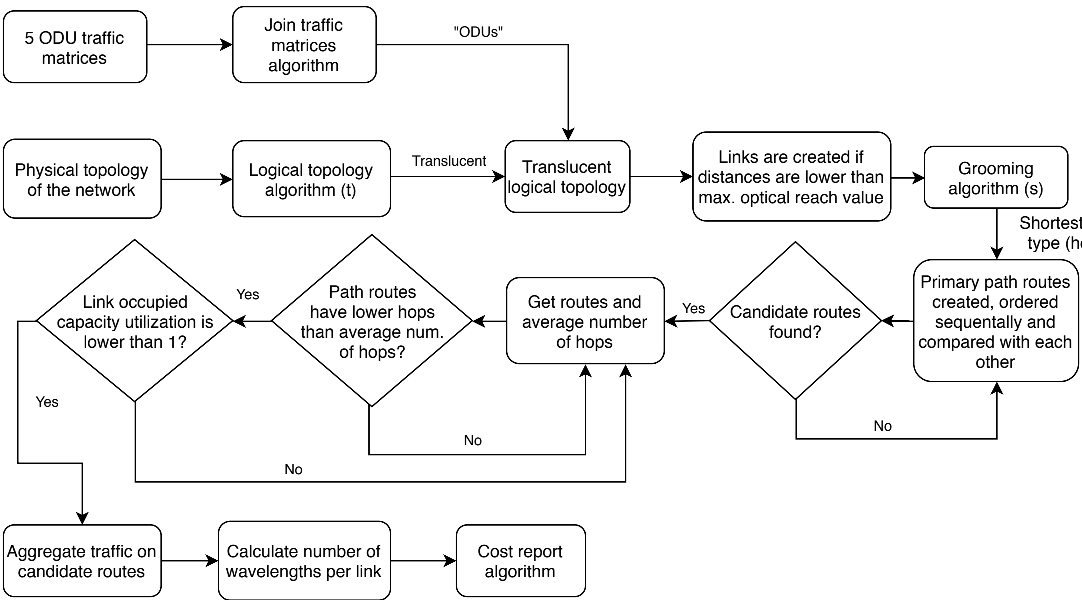

Figura 2.43: Fluxogram with the steps performed in the translucent without survivability transport mode approach.

Creation and join the traffic matrices

The first step is to create the traffic matrices based on the reference network **??**. In order to create the 5 traffic matrices in Net2Plan it is necessary the length of all the links and the total traffic used in this network, so later it is needed to define in Net2Plan the length in all end nodes and the total traffic depends on the value of traffic used (low traffic - 0.5 Tbit/s, medium traffic - 5 Tbit/s and high traffic - 10 Tbit/s). As you can see in the figure below, it is defined the path of the 5 ODUs and they will be aggregated in just one single ODU, making it possible to join all the demands in just one file and load it later into the network. This final resulting ODU joins the multiple traffic demands from all the traffic matrices previously created and, of course, the traffic demands will depend on the values used on the creation of the matrices (low, medium and high traffic).

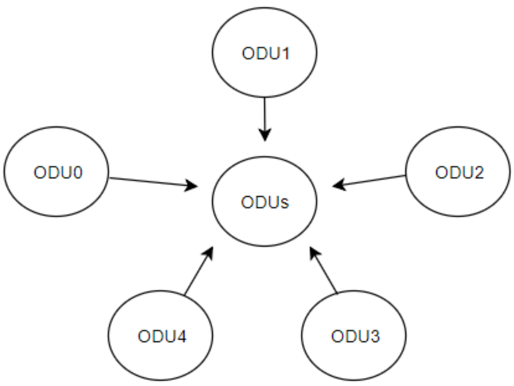

Figura 2.44: Join the 5 ODU traffic matrices into 1 single file "ODUs". The 5 traffic demands from the traffic matrices previously created are joined into 1 file to load it later on Net2Plan.

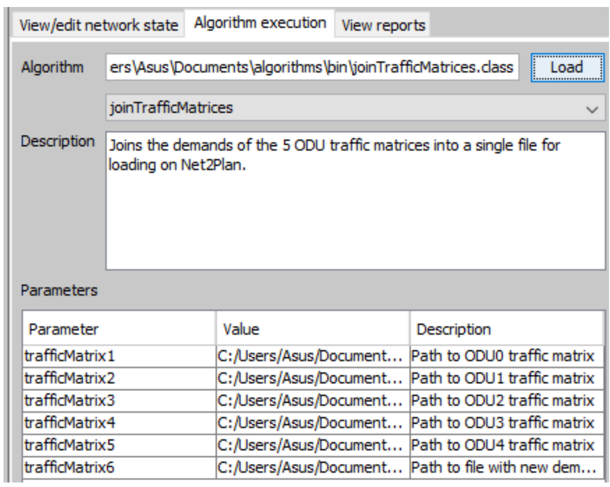

Figura 2.45: Load of the join traffic matrices algorithm for the translucent transport mode on Net2Plan. It is defined the 5 paths to load the 5 ODU traffic matrices and the last path is the one where will be saved the file that joins all 5 the traffic demands.

Creation of the physical topology

The next step is to create the allowed physical topology of the network in Net2Plan. This network consists in 6 nodes and 8 bidirectional links. It is now also possible to define the length in all links. In the figure below it is shown the allowed physical topology in this transport mode.

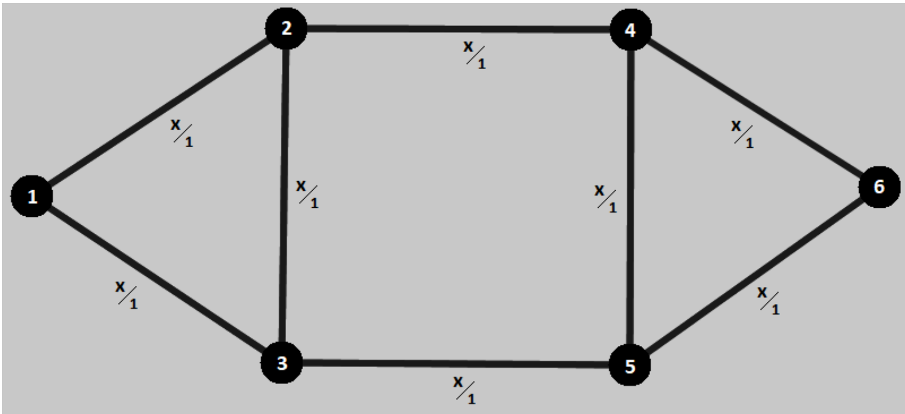

Figura 2.46: Allowed physical topology. The allowed physical topology is defined by the duct and sites in the field. It is assumed that each duct supports up to 1 bidirectional transmission system and each site supports up to 1 node.

Creation of the logical topology

It is now time to create the allowed logical topology. A network topology represents how the links and the nodes of the network interconnect with each other and the logical topology algorithm creates the logical topology on another layer. In the translucent transport mode each node connects to other node if the shortest path distance between them is lower than the maximum optical reach value (in km) that an optical signal can traverse in an optical channel. If the maximum optical reach is higher than the shortest path of a link, then it is not created a logical link of that specific physical link. On the contrary, there are created direct links between all node pairs in the network that follow this rule. These additions of links between end nodes are made in the new upper layer of the network. The respective demands are saved in the new upper layer and those demands from the lower layer are then removed. The lower layer is the physical layer of the network and it is now created a new upper layer which is the logical layer of the network and represents the logical topology of the translucent transport mode. The allowed physical and optical topologies, the logical topologies for all ODUs and the resulting physical topology is shown in the next section below 2.7.13 for the three traffic scenarios. It is shown below three figures with the code in Java of the creation of the network logical topology, the load of the logical topology algorithm in Net2Plan and the resulting allowed optical topology for the translucent transport mode without survivability.

```java
if (netPlan.isSingleLayer() && logicalTopology.equalsIgnoreCase("Translucent")) {

    int maximumOpticalReach = Integer.parseInt(algorithmParameters.get("maximumOpticalReach"));
    maxOpticalReach = maximumOpticalReach;

    sendToFile("opticalReach.txt");

    this.lowerLayer = netPlan.getNetworkLayerDefault();
    lowerLayer.setName("Physical Topology");
    this.upperLayer = netPlan.addLayer("Logical Topology Translucent","Upper layer of the design","ODU","ODU",null);
    upperLayer.setDescription("Translucent Logical Topology"+" - Maximum Optical Reach= "+maximumOpticalReach+" km");
    netPlan.removeAllLinks(upperLayer);

    for (Node i : netPlan.getNodes()) {
        for (Node j : netPlan.getNodes()) {
            if (i.getIndex() != j.getIndex()) {
                if (netPlan.getNodePairEuclideanDistance(i, j) <= maximumOpticalReach) {
                    netPlan.addLink(i, j, 0, netPlan.getNodePairEuclideanDistance(i, j), 200000, null, upperLayer);
                }
            }
        }
    }
}
```

Figura 2.47: Java code of the logical topology approach for the translucent transport mode. The logical layer is created by adding direct links between all end nodes if their shortest path between them is lower than the maximum optical reach value. The new layer is now the translucent logical topology of the network.

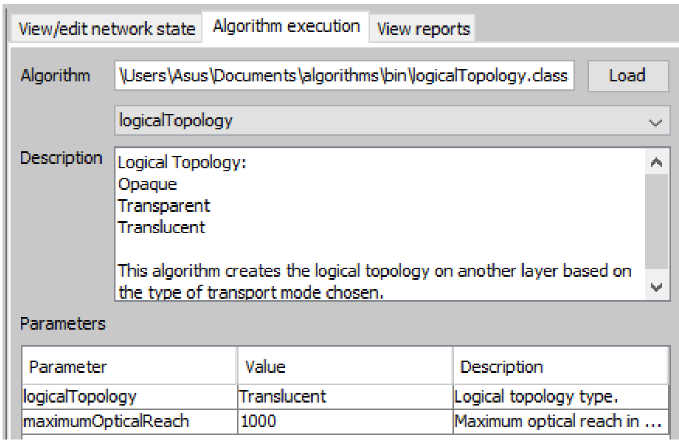

Figura 2.48: Load of the logical topology algorithm for the translucent transport mode on Net2Plan. It is assumed that the maximum optical reach value is 1000 km, so all direct links between node pairs with different index are created in all optical channels.

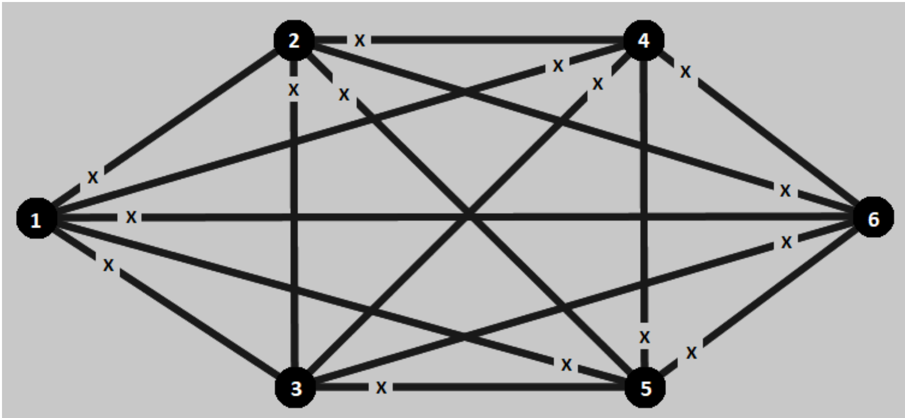

Figura 2.49: Allowed optical topology. It is assumed that each connections between demands supports up to 100 lightpaths. In the translucent transport mode there are several possible logical topologies for a specific network. It depends, for example, on the maximum optical reach value that represents the maximum distance that an optical signal can traverse a link without the need of regenerate.

Creation of routes and aggregation of traffic

After a network topology is created, it is now time to set the routing and then grooming algorithms. In the translucent without survivability transport mode the objective function is to minimize the lightpaths of the network and to find a route for every lightpath demand without considering the wavelength assignment. In this heuristic approach the goal is to minimize the lightpaths blocked and going through the carried lightpaths search for the ones that minimizes the average number of physical hops. Some of the links in the network will suffer from excessive traffic and some other links will occupy only a small part of the bandwidth. In order to do more efficiently, the link's utilizations are almost 1 so there is less non occupied capacity. Firstly, the candidate routes are found by the Dijkstra algorithm for each pair of nodes and calculates the minimum distance for each source and destination nodes and compares with each other. Then, the wavelength assignments are carried out for every lightpaths that have the minimum average of hops for each route and if the path distance is lower than the maximum optical reach value (in this report the maximum optical reach value used is 1000 km). The carried lightpaths resulted previously are sequentially processed.

```
case "Logical Topology Translucent":

    int hops = 0;
    Set<Route> nRoutes = new HashSet<Route>();

    for (Demand d : netPlan.getDemands(lowerLayer)) {

        nRoutes = d.getRoutes();
        for (Route c : nRoutes) {
            hops += c.getNumberOfHops();
        }
    }
    int n = hops/netPlan.getNumberOfRoutes(lowerLayer);

    for (Demand d : netPlan.getDemands(lowerLayer)) {
        boolean odd = true;
        int counter = 0;

        Set<Route> droutes = d.getRoutes();

        for (Route c : droutes) {
            counter++;
            boolean jump = false;

            if (odd) {
                if (c.getNumberOfHops() < (n-1) && c.getLengthInKm() <= maxOpticalReach) {
                    c.setCarriedTraffic(d.getOfferedTraffic(), d.getOfferedTraffic());
                    save = c;
                    System.out.println("Roots");
                }
                else if (c.getNumberOfHops() > (n-1) && c.getLengthInKm() <= maxOpticalReach) {
                    save = c;
                    System.out.println("Roots");
                }
```

Figura 2.50: Creation of routes and aggregation of traffic for the translucent without survivability transport mode. The candidate routes are searched by the shortest path type method and the average minimum number of physical hops of the routes. The offered traffic demands are set into these routes.

```
for (long tNodeId : tNodeIdsTransl) {
    inTransl = netPlan.getNodeFromId(tNodeId);
    for (long tNodeId1 : tNodeIdsTransl) {

        if (tNodeId == tNodeId1)
            continue;

        outTransl = netPlan.getNodeFromId(tNodeId1);
        double totaltraffic = 0;

        groomRouteTransl = netPlan.getNodePairRoutes(inTransl, outTransl, false, lowerLayer);
        protectRoutesTransl = netPlan.getNodePairProtectionSegments(inTransl, outTransl, false, lowerLayer);

        for (Route c : groomRouteTransl) {
            if(c.getCarriedTraffic() <= c.getOccupiedCapacity()){
                compareTransl = c;
            }
        }

        pathTransl = compareTransl.getSeqLinksRealPath();

        for(Route c : groomRouteTransl){
            for (Link link : pathTransl){
                if(link.getUtilizationNotIncludingProtectionSegments() < 1){
                    totaltraffic = totaltraffic + c.getCarriedTraffic();
                    compareTransl2 = c;
                }
                else
                    compareTransl = c;
            }
        }

        pathTransl2 = compareTransl2.getSeqLinksRealPath();
```

Figura 2.51: Creation of routes and aggregation of traffic for the translucent without survivability transport mode. Minimizing the blocked traffic, the traffic demands are set into the candidate primary path routes found earlier.

Function	Definition
netPlan.getDemands(lowerLayer)	Returns the array of demands for the lower layer.
d.getRoutes()	Returns all the routes associated to the demand "d".
c.getNumberOfHops()	Returns the route number of traversed links.
c.getLengthInKm()	Returns the route length in km, summing the traversed link lengths, as many times as the link is traversed.
c.setCarriedTraffic()	Sets the route carried traffic and the occupied capacity in the links, setting it up to be the same in all links.
d.getOfferedTraffic()	Returns the offered traffic of the demand "d".
netPlan.getNodeIds()	Returns the array of the nodes' indexes.
netPlan.getNodeFromId(tNodeId)	Returns the node with the index "tNodeId".
netPlan.getNodePairRoutes (in,out,false,lowerLayer)	Returns the routes at "lowerLayer" from nodes "in" and "out".
c.getCarriedTraffic()	Returns the route carried traffic at that moment.
c.getOccupiedCapacity()	Returns the route occupied capacity at that moment.

Tabela 2.118: Table with the description of the main functions in the creation of routes and aggregation of traffic in the grooming algorithm.

Calculation of the number of wavelengths per link

The final step of the routing and grooming algorithms is to calculate the number of wavelengths per link for the whole network. This is the last and an important step because with the number of wavelengths per link in the network, it is possible to calculate other network components. In the translucent transport mode, as in the figure below shows, the algorithm starts with going through all the nodes which have different index between them (end nodes) and if the path distance between them is lower than the maximum optical reach value and in all the links that crosses between these pairs of nodes is reserved a link capacity based on the previous traffic aggregation on routes. The total carried traffic in the link including protection and non-protection segments will be divided by the wavelength capacity and it is now possible to obtain the number of wavelengths per link.

```
for (Link link:path)
{
    String nw = link.getAttribute("nW");
    nW=0;

    if(nw!=null)
    {
        nW = Integer.parseInt(nw);
        nW = (int) (nW+Math.ceil(totaltraffic/wavelengthCapacity));
        link.setAttribute("nW",String.valueOf(nW));
    }else {
        nW = (int) Math.ceil(totaltraffic/wavelengthCapacity);
        link.setAttribute("nW",String.valueOf(nW));
    }
}
```

Figura 2.52: Calculation of the number of wavelengths per link for the translucent transport mode. The link capacity is reserved based on the previous traffic aggregation.

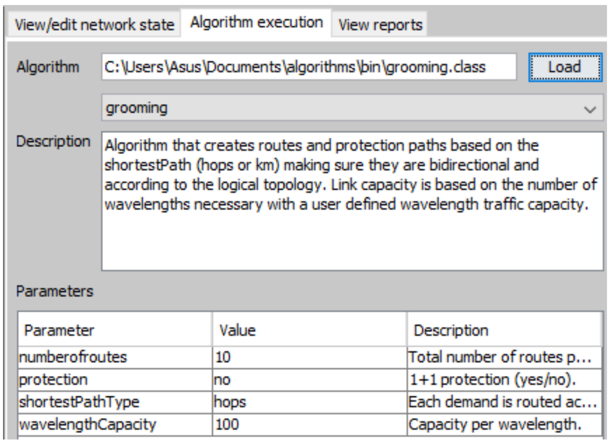

Figura 2.53: Load of the grooming algorithm. The total number of routes per demand is set to 10, the user can define if the model is with or without protection, the shortest path type is set to "hops" and the capacity per wavelength is used 100 optical channels.

Network cost report

In order to obtain the network CAPEX results, the formulas needed to calculate the network elements and that are demonstrated previously in the beginning of this section 2.7.13 were "translated" into Java code in a cost report algorithm. This algorithm can be loaded in Net2Plan and calculates and shows in tables the network CAPEX and also the per-link and per-node information with more details.

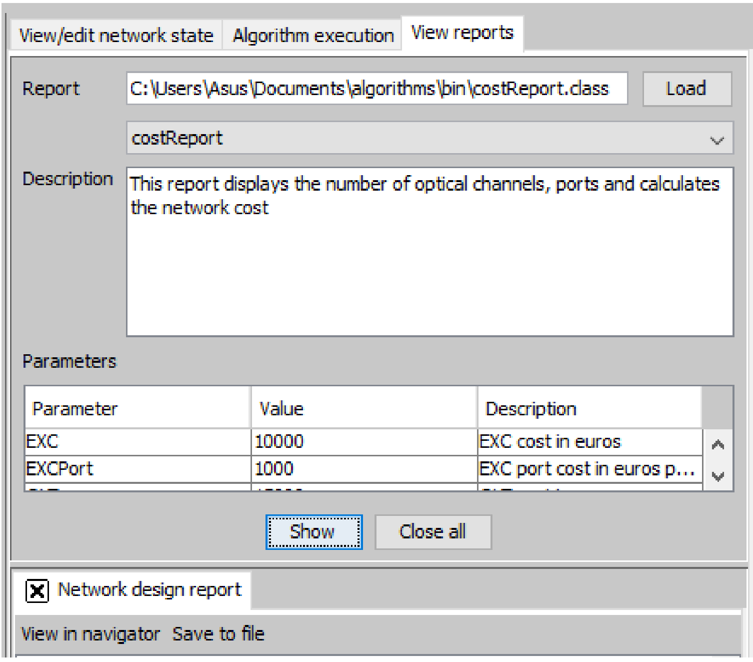

Figura 2.54: Load of the cost report algorithm on Net2Plan. The result view is an HTML page with the network optical and electrical components and their costs.

Result description

It is already known all the necessary formulas to obtain the CAPEX value for the reference network **??**. As described in the subsection of the network traffic **??**, it is necessary to obtain three different values of CAPEX for the low (0.5 Tbit/s), medium (5 Tbit/s) and high (10 Tbit/s) traffic. It is used a network software program called Net2Plan which can design the traffic matrices, create all the network topologies, simulate the algorithms into the network implemented in the programming software called Eclipse and analyze the results obtained. In this chapter will be demonstrated the results by Pedro's heuristics. In each of the three traffic scenarios, it will be shown the network topologies followed by the table with the CAPEX value of the network.

Low Traffic Scenario:

In this scenario we have to take into account the traffic calculated in **??**. In a first phase we will show the various existing topologies of the network. The first are the allowed topologies, physical and optical topologies, the second are the logical topology for all ODUs and finally the resulting physical topology.

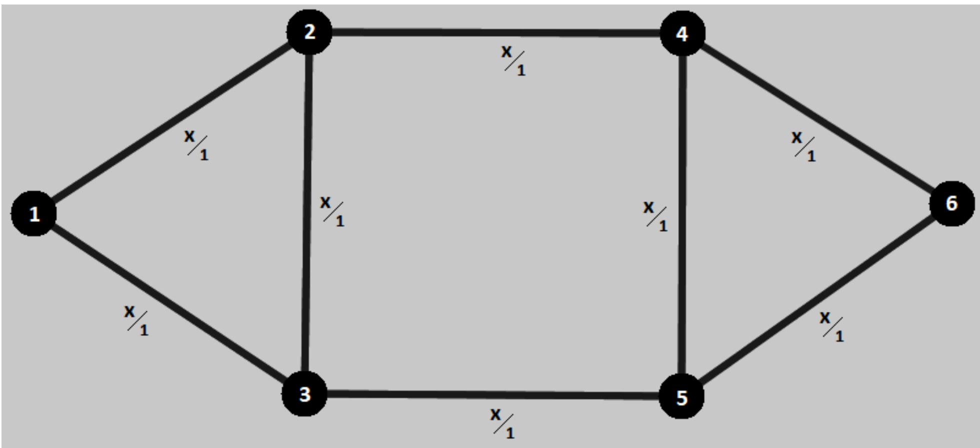

Figura 2.55: Allowed physical topology. The allowed physical topology is defined by the duct and sites in the field. It is assumed that each duct supports up to 1 bidirectional transmission system and each site supports up to 1 node.

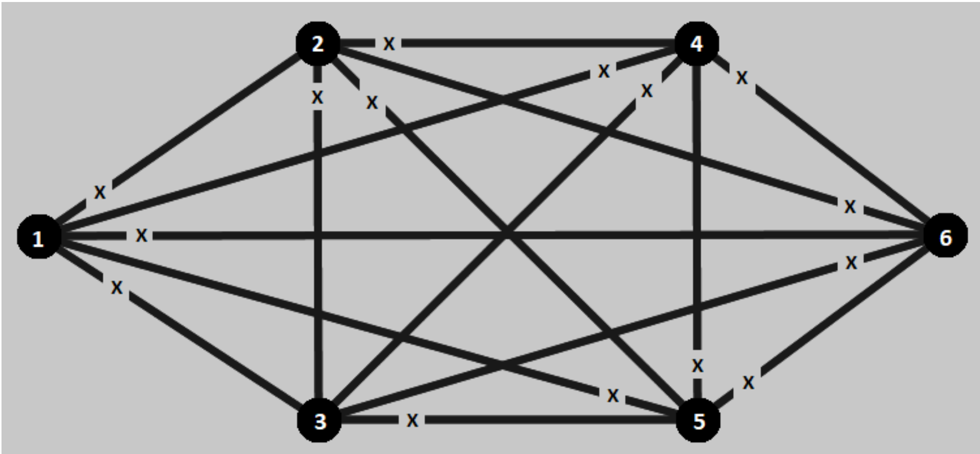

Figura 2.56: Allowed optical topology. The allowed optical topology is defined by the transport mode (translucent transport mode in this case). It is assumed that each connections between demands supports up to 100 lightpaths.

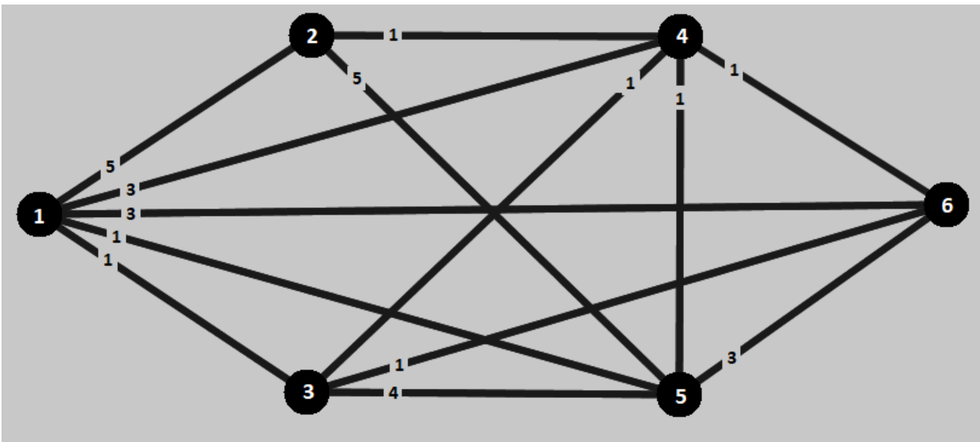

Figura 2.57: ODU0 logical topology defined by the ODU0 traffic matrix.

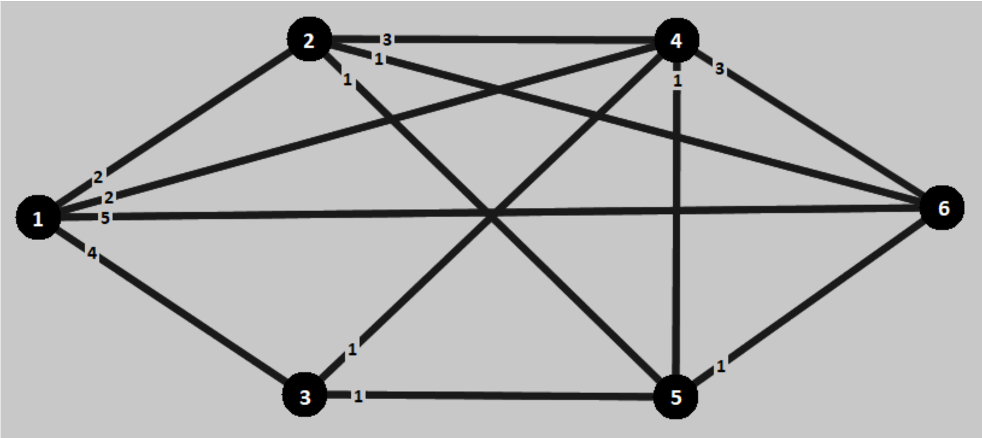

Figura 2.58: ODU1 logical topology defined by the ODU1 traffic matrix.

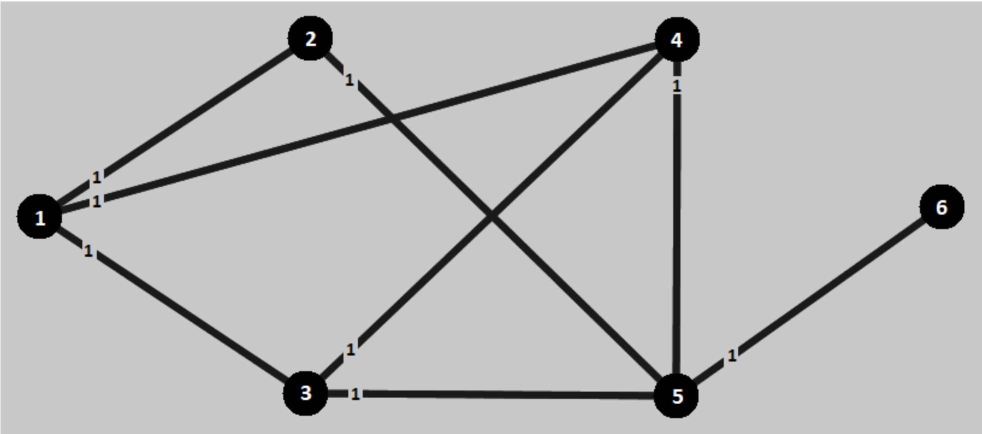

Figura 2.59: ODU2 logical topology defined by the ODU2 traffic matrix.

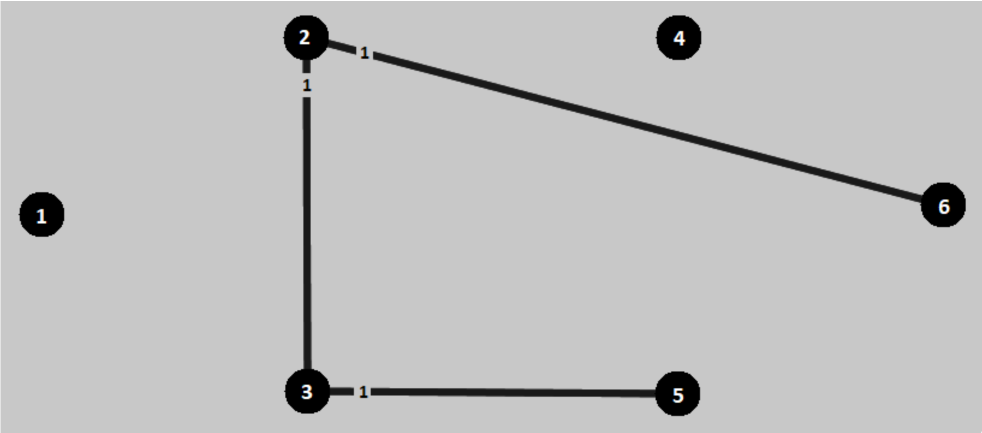

Figura 2.60: ODU3 logical topology defined by the ODU3 traffic matrix.

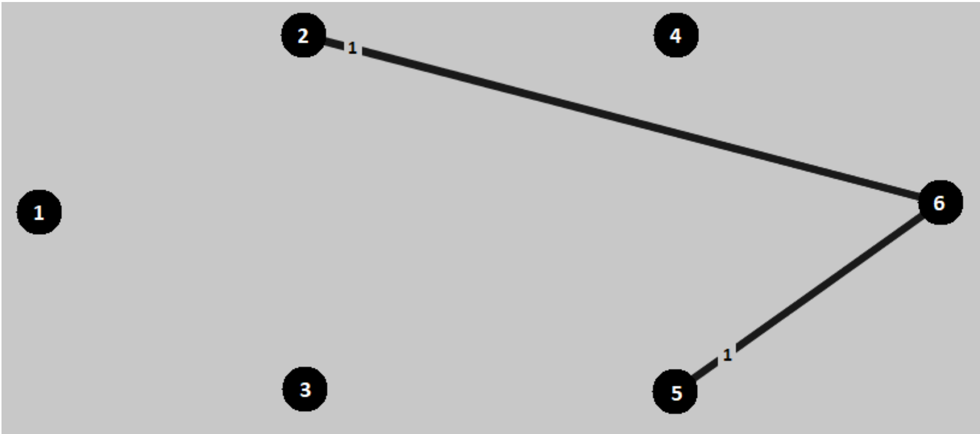

Figura 2.61: ODU4 logical topology defined by the ODU4 traffic matrix.

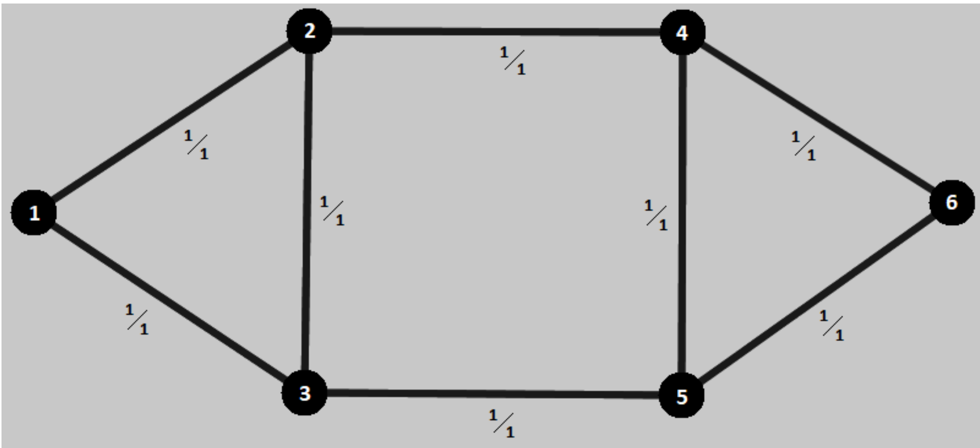

Figura 2.62: Physical topology after dimensioning.

Following all the steps mentioned in the **??**, applying the routing and grooming heuristic algorithms in the Net2Plan software and using all the data referring to this scenario, the obtained result for the Pedro's heuristics can be consulted in the following table 2.119.

CAPEX of the Network						
			Quantity	Unit Price	Cost	Total
Link Cost	OLTs		16	15 000 €	240 000 €	9 520 000 €
	100 Gbits/s Transceivers		18	5 000 €/Gbit/s	9 000 000 €	
	Amplifiers		70	4 000 €	280 000 €	
Node Cost	Electrical	EXCs	6	10 000 €	60 000 €	2 072 590 €
		ODU0 Ports	60	10 €/port	600 €	
		ODU1 Ports	50	15 €/port	750 €	
		ODU2 Ports	16	30 €/port	480 €	
		ODU3 Ports	6	60 €/port	360 €	
		ODU4 Ports	4	100 €/port	400 €	
		Transponders	18	100 000 €/port	1 800 000 €	
	Optical	OXCs	6	20 000 €	120 000 €	
		Line Ports	18	2 500 €/port	45 000 €	
		Add Ports	18	2 500 €/port	45 000 €	
Total Network Cost						11 592 590 €

Tabela 2.119: Table with detailed description of CAPEX.

All the values calculated in the previous table were obtained through the equations 1.2 and 1.3 referred to in section 1.1, but for a more detailed analysis we created table 2.120 where we can see how all the parameters are calculated individually.

Medium Traffic Scenario:

In this scenario we have to take into account the traffic calculated in **??**. In a first phase we will show the various existing topologies of the network. The first are the allowed topologies, physical and optical topologies, the second are the logical topology for all ODUs and finally the resulting physical topology.

	Equation used to calculate the cost
OLTs	$2\sum_{i=1}^{N}\sum_{j=i+1}^{N} L_{ij}\gamma_0^{OLT}$
Transceivers	$2\sum_{i=1}^{N}\sum_{j=i+1}^{N} L_{ij}f_{ij}^{od}\gamma_1^{OLT}\tau$
Amplifiers	$2\sum_{i=1}^{N}\sum_{j=i+1}^{N} L_{ij}N_{ij}^{R}c^{R}$
EXCs	$\sum_{n=1}^{N} N_{exc,n}\gamma_{e0}$
ODU0 Port	$\sum_{n=1}^{N}\sum_{d=1}^{N} N_{exc,n}D_{nd,0}\gamma_{e1,0}$
ODU1 Port	$\sum_{n=1}^{N}\sum_{d=1}^{N} N_{exc,n}D_{nd,1}\gamma_{e1,1}$
ODU2 Port	$\sum_{n=1}^{N}\sum_{d=1}^{N} N_{exc,n}D_{nd,2}\gamma_{e1,2}$
ODU3 Port	$\sum_{n=1}^{N}\sum_{d=1}^{N} N_{exc,n}D_{nd,3}\gamma_{e1,3}$
ODU4 Port	$\sum_{n=1}^{N}\sum_{d=1}^{N} N_{exc,n}D_{nd,4}\gamma_{e1,4}$
LR Transponders	$\sum_{n=1}^{N}\sum_{j=1}^{N} N_{exc,n}\lambda_{od}\gamma_{e1,-1}$
OXCs	$\sum_{n=1}^{N} N_{oxc,n}\gamma_{o0}$
Add Port	$\sum_{n=1}^{N}\sum_{j=1}^{N} N_{oxc,n}\lambda_{od}\gamma_{o1}$
Line Port	$\sum_{n=1}^{N}\sum_{j=1}^{N} N_{oxc,n}f_{ij}^{od}\gamma_{o1}$
CAPEX	The final cost is calculated by summing all previous results.

Tabela 2.120: Table with description of calculation

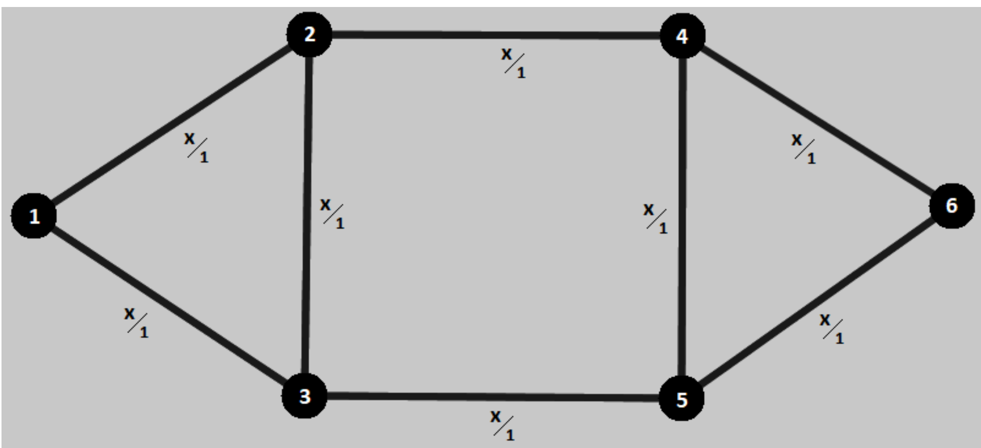

Figura 2.63: Allowed physical topology. The allowed physical topology is defined by the duct and sites in the field. It is assumed that each duct supports up to 1 bidirectional

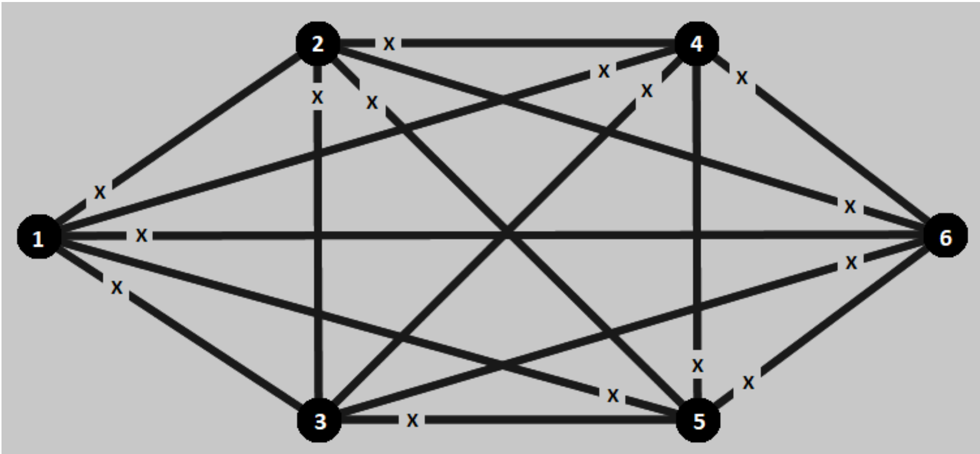

Figura 2.64: Allowed optical topology. The allowed optical topology is defined by the transport mode (translucent transport mode in this case). It is assumed that each connections between demands supports up to 100 lightpaths.

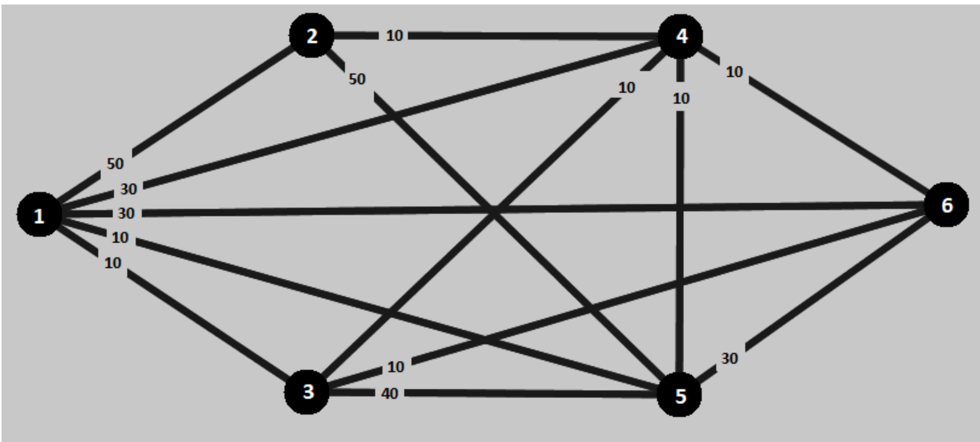

Figura 2.65: ODU0 logical topology defined by the ODU0 traffic matrix.

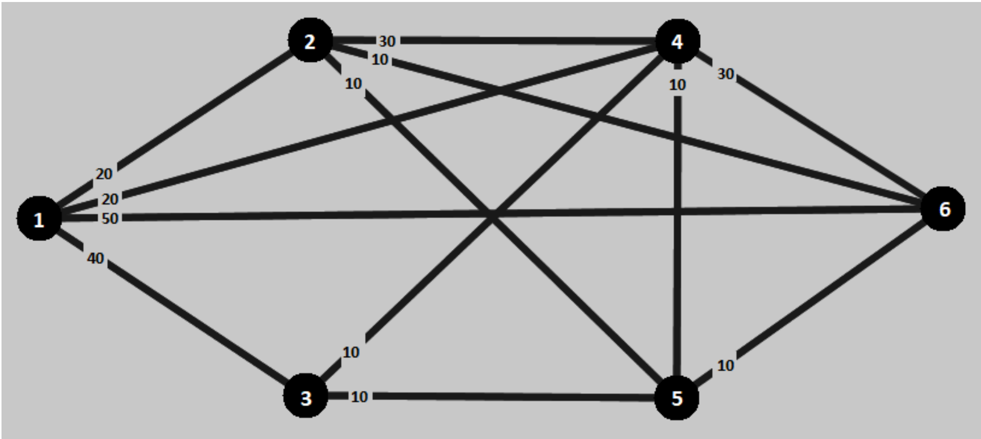

Figura 2.66: ODU1 logical topology defined by the ODU1 traffic matrix.

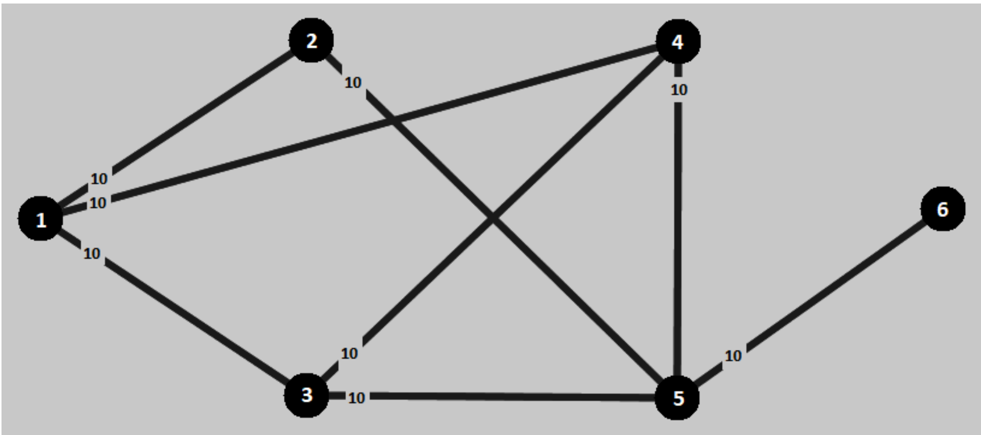

Figura 2.67: ODU2 logical topology defined by the ODU2 traffic matrix.

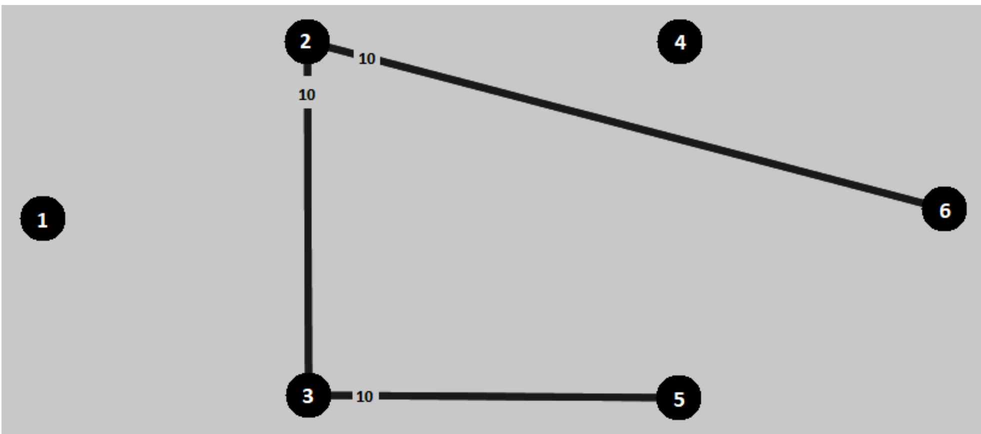

Figura 2.68: ODU3 logical topology defined by the ODU3 traffic matrix.

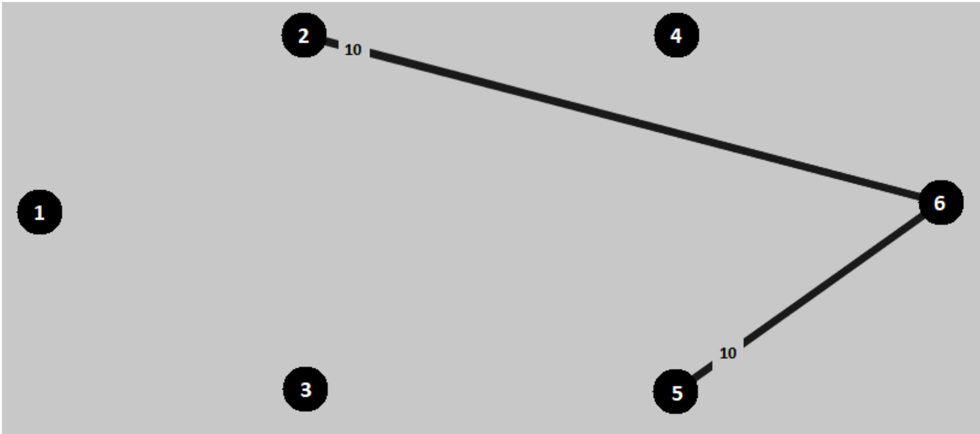

Figura 2.69: ODU4 logical topology defined by the ODU4 traffic matrix.

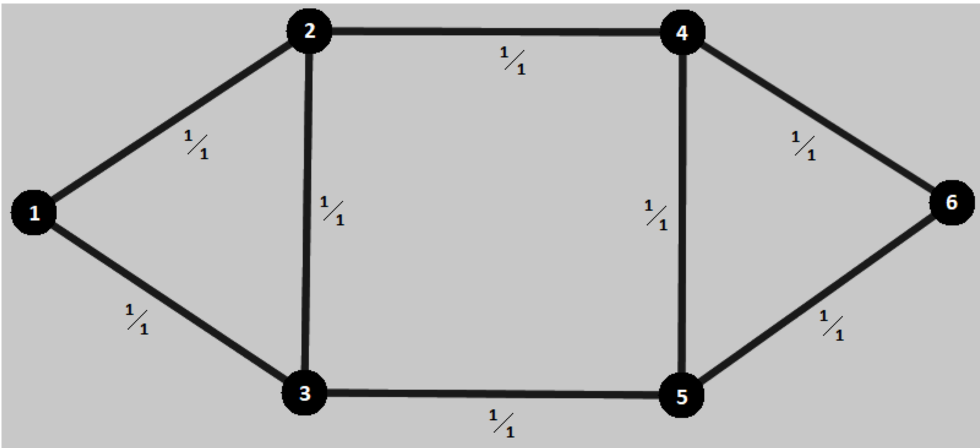

Figura 2.70: Physical topology after dimensioning.

Following all the steps mentioned in the **??**, applying the routing and grooming heuristic algorithms in the Net2Plan software and using all the data referring to this scenario, the obtained result for the Pedro's heuristics can be consulted in the following table 2.121. In table 2.120 mentioned in previous scenario we can see how all the values were calculated.

CAPEX of the Network						
			Quantity	Unit Price	Cost	Total
Link Cost	OLTs		16	15 000 €	240 000 €	40 520 000 €
	100 Gbits/s Transceivers		80	5 000 €/Gbit/s	40 000 000 €	
	Amplifiers		70	4 000 €	280 000 €	
Node Cost	Electrical	EXCs	6	10 000 €	60 000 €	8 605 900 €
		ODU0 Ports	600	10 €/port	6 000 €	
		ODU1 Ports	500	15 €/port	7 500 €	
		ODU2 Ports	160	30 €/port	4 800 €	
		ODU3 Ports	60	60 €/port	3 600 €	
		ODU4 Ports	40	100 €/port	4 000 €	
		Transponders	80	100 000 €/port	8 000 000 €	
	Optical	OXCs	6	20 000 €	120 000 €	
		Line Ports	80	2 500 €/port	200 000 €	
		Add Ports	80	2 500 €/port	200 000 €	
Total Network Cost						49 125 900 €

Tabela 2.121: Table with detailed description of CAPEX.

High Traffic Scenario:

In this scenario we have to take into account the traffic calculated in **??**. In a first phase we will show the various existing topologies of the network. The first are the allowed topologies, physical and optical topologies, the second are the logical topology for all ODUs and finally the resulting physical topology.

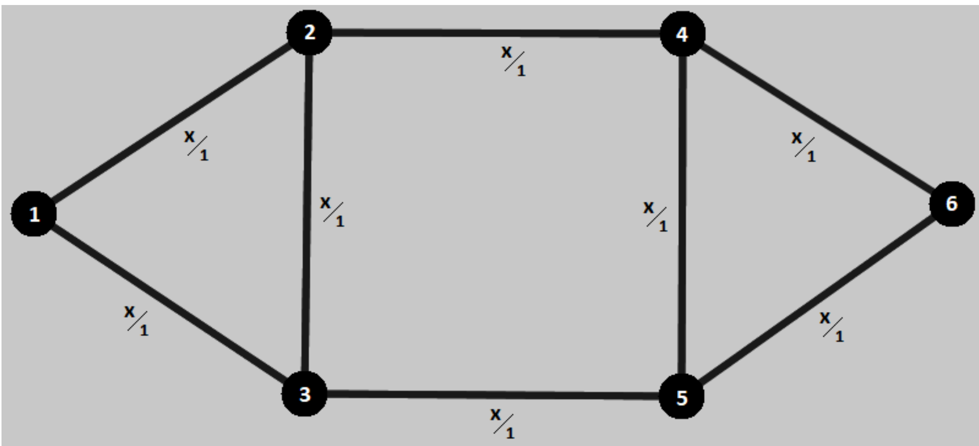

Figura 2.71: Allowed physical topology. The allowed physical topology is defined by the duct and sites in the field. It is assumed that each duct supports up to 1 bidirectional transmission system and each site supports up to 1 node.

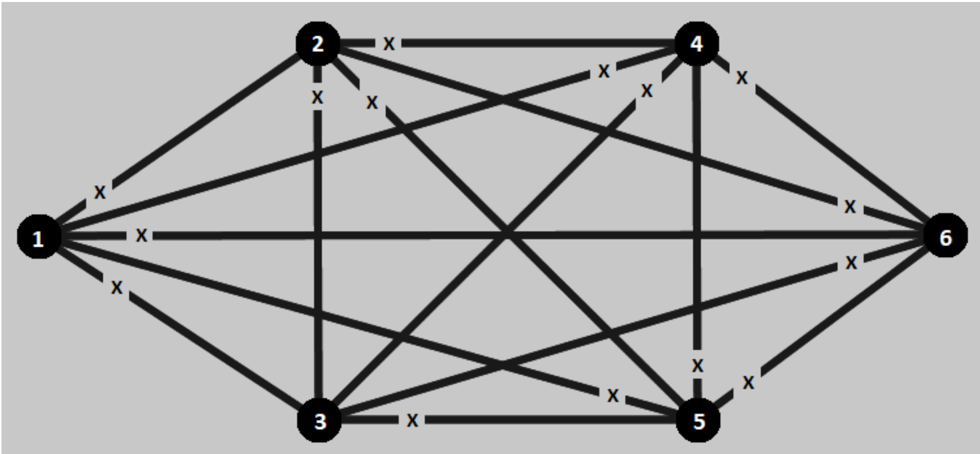

Figura 2.72: Allowed optical topology. The allowed optical topology is defined by the transport mode (translucent transport mode in this case). It is assumed that each connections between demands supports up to 100 lightpaths.

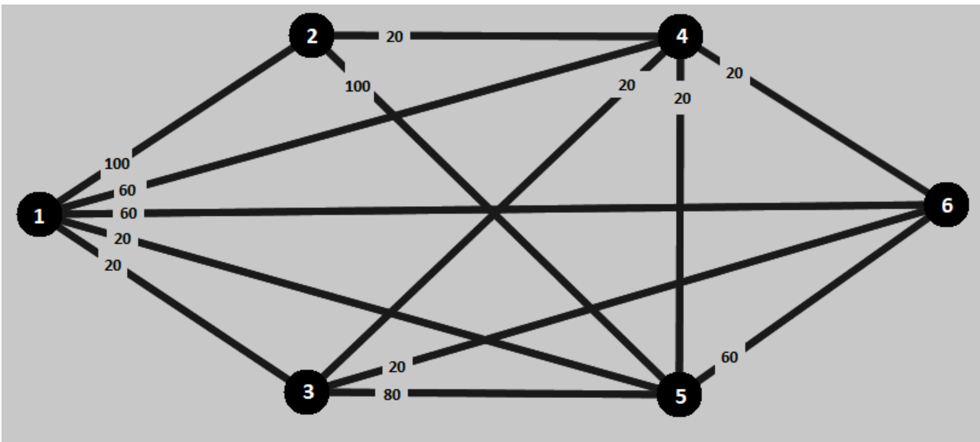

Figura 2.73: ODU0 logical topology defined by the ODU0 traffic matrix.

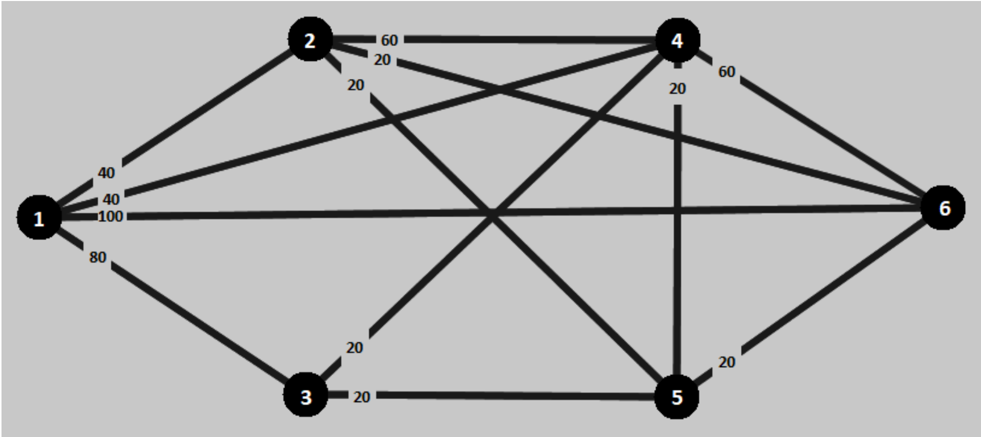

Figura 2.74: ODU1 logical topology defined by the ODU1 traffic matrix.

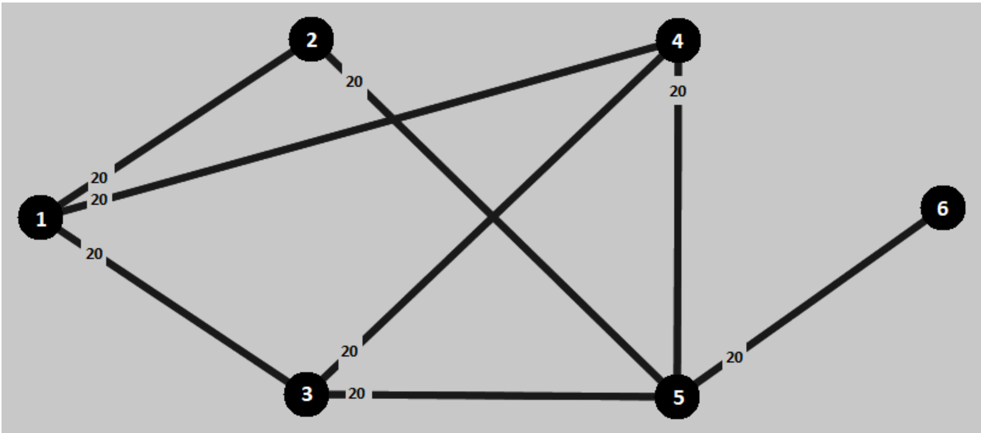

Figura 2.75: ODU2 logical topology defined by the ODU2 traffic matrix.

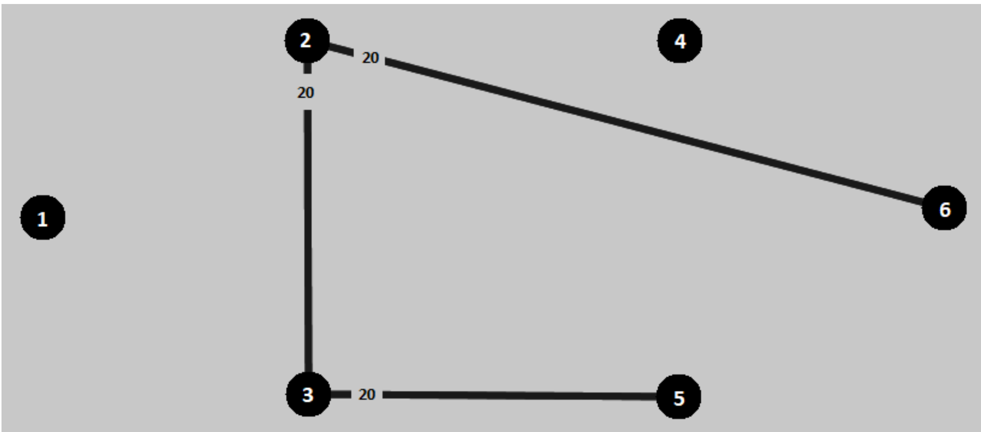

Figura 2.76: ODU3 logical topology defined by the ODU3 traffic matrix.

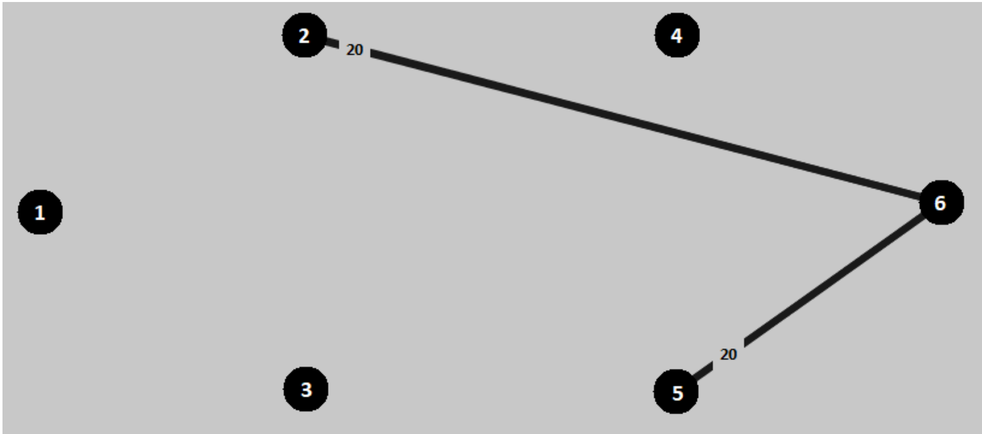

Figura 2.77: ODU4 logical topology defined by the ODU4 traffic matrix.

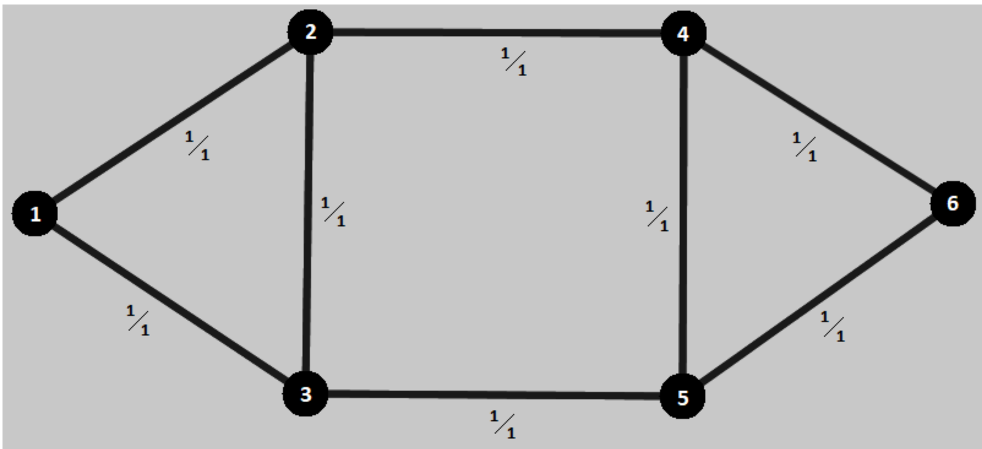

Figura 2.78: Physical topology after dimensioning.

Following all the steps mentioned in the **??**, applying the routing and grooming heuristic algorithms in the Net2Plan software and using all the data referring to this scenario, the obtained result for the Pedro's heuristics can be consulted in the following table 2.122. In table 2.120 mentioned in previous scenario we can see how all the values were calculated.

<table>
<tr><th colspan="7">CAPEX of the Network</th></tr>
<tr><th colspan="3"></th><th>Quantity</th><th>Unit Price</th><th>Cost</th><th>Total</th></tr>
<tr><td rowspan="3">Link Cost</td><td colspan="2">OLTs</td><td>16</td><td>15 000 €</td><td>240 000 €</td><td rowspan="3">77 520 000 €</td></tr>
<tr><td colspan="2">100 Gbits/s Transceivers</td><td>154</td><td>5 000 €/Gbit/s</td><td>61 000 000 €</td></tr>
<tr><td colspan="2">Amplifiers</td><td>70</td><td>4 000 €</td><td>280 000 €</td></tr>
<tr><td rowspan="10">Node Cost</td><td rowspan="7">Electrical</td><td>EXCs</td><td>6</td><td>10 000 €</td><td>60 000 €</td><td rowspan="10">16 401 800 €</td></tr>
<tr><td>ODU0 Ports</td><td>1 200</td><td>10 €/port</td><td>12 000 €</td></tr>
<tr><td>ODU1 Ports</td><td>1 000</td><td>15 €/port</td><td>15 000 €</td></tr>
<tr><td>ODU2 Ports</td><td>320</td><td>30 €/port</td><td>9 600 €</td></tr>
<tr><td>ODU3 Ports</td><td>120</td><td>60 €/port</td><td>7 200 €</td></tr>
<tr><td>ODU4 Ports</td><td>80</td><td>100 €/port</td><td>8 000 €</td></tr>
<tr><td>Transponders</td><td>154</td><td>100 000 €/port</td><td>15 400 000 €</td></tr>
<tr><td rowspan="3">Optical</td><td>OXCs</td><td>6</td><td>20 000 €</td><td>120 000 €</td></tr>
<tr><td>Line Ports</td><td>154</td><td>2 500 €/port</td><td>385 000 €</td></tr>
<tr><td>Add Ports</td><td>154</td><td>2 500 €/port</td><td>385 000 €</td></tr>
<tr><td colspan="6">Total Network Cost</td><td>93 921 800 €</td></tr>
</table>

Tabela 2.122: Table with detailed description of CAPEX.

Conclusions

Once we have obtained the results for all the scenarios we will now draw some conclusions about these results. For a better analysis of the results will be created the table 2.123 with the number of line ports, tributary ports and transceivers because they are important values for the cost of CAPEX, the cost of links, the cost of nodes and finally the cost of CAPEX.

	Low Traffic	Medium Traffic	High Traffic
Traffic (Gbit/s)	500	5 000	10 000
Bidirectional Links used	8	8	8
Number of Add ports	18	80	154
Number of Line ports	18	80	154
Number of Tributary ports	136	1 360	2 720
Number of Transceivers	18	80	154
Link Cost	9 520 000 €	40 520 000 €	77 520 000 €
Node Cost	2 072 590 €	8 605 900 €	16 401 800 €
CAPEX	**11 592 590 €**	**49 125 900 €**	**93 921 800 €**
CAPEX/Gbit/s	**23 185 €/Gbit/s**	**9 825 €/Gbit/s**	**9 392 €/Gbit/s**

Tabela 2.123: Table with different value of CAPEX for this case.

Looking at the previous table we can make some comparisons between the several scenario:

- Comparing the low traffic with the others we can see that despite having an increase of factor ten (medium traffic) and factor twenty (high traffic), the same increase does not occur in the final cost (it is lower);

  This happens because the number of the transceivers is lower than expected which leads by carrying the traffic with less network components and, consequently, the network CAPEX is lower;

- Comparing the medium traffic with the high traffic we can see that the increase of the factor is double and in the final cost this factor is very close but still inferior;

  This happens because the number of the transceivers is also lower but very close to the expected;

- Comparing the CAPEX cost per bit we can see that in the low traffic the cost is higher than the medium and high traffic, which in these two cases the value is similar, but still inferior in the higher traffic. The difference between the low traffic and medium/high traffics is significantly high;

  This happens because the lower the traffic, the higher CAPEX/bit will be. We can see that in medium and high traffic the results tend to be one closer and lower value.

Opens Issues

The creation of this model for any scenario, started with some considerations and some open issues being:

- Allow maximum optical reach value in optical channels.

  It is assumed that the maximum optical reach value is 1000 km, so all direct links between node pairs with different index are created in all optical channels;

- Allow multiple transmission system.

  The presented model for each link only supports one transmission system.

2.7.14 Translucent with 1+1 Protection

Student Name	:	Pedro Coelho (01/03/2018 -)
Goal	:	Implement the heuristic model for the translucent transport mode with 1 plus 1 protection.

The translucent networks in the translucent transport mode with 1+1 protection consist also of intermediate optical network architectures between opaque and transparent networks like this transport mode without survivability. In this, there will be created backup paths in order to prevent a network failure and loose all traffic data. The methodology is the same but in this case if, for any reason, there is a link or a path that do not work, then the traffic data can traverse to the same destination path in different links that were created for this purpose. These new created paths are longer than the primary ones and the network CAPEX will be significantly higher. The OEO conversions are also made when the optical signal falls and the maximum optical reach value is used to regenerate the same optical signal.

For cost savings, the translucent networks aims at using the minimum number of regenerators and wavelengths of the network, being the most advantageous transport mode in the optical backbone networks.

We also must take into account the following particularity of this mode of transport:

- $N_{OXC,n}$ = 1, $\forall$ n that process traffic
- $N_{EXC,n}$ = 1, $\forall$ n that process traffic

The minimization of the network CAPEX is made through the equation 1.1 where in this case for the cost of nodes we have in consideration the electric cost 1.4 and the optical cost 1.5.

In this case the value of $P_{exc,c,n}$ is obtained by equation 2.7 for short-reach and by the equation 2.8 for long-reach and the value of $P_{oxc,n}$ is obtained by equation 2.9.

The equation 2.7 refers to the number of short-reach ports of the electrical switch with bit-rate c in node n, $P_{exc,c,n}$, i.e. the number of tributary ports with bit-rate c in node n which can be calculated as

$$P_{exc,c,n} = \sum_{d=1}^{N} D_{nd,c} \tag{2.7}$$

where $D_{nd,c}$ are the client demands between nodes n and d with bit rate c.

In this case there is the following particularity:

- When n=d the value of client demands is always zero, i.e, $D_{nn,c} = 0$

As previously mentioned, the equation 2.8 refers to the number of long-reach ports of the electrical switch with bit-rate -1 in node n, $P_{exc,-1,n}$, i.e. the number of add ports of node n which can be calculated as

$$P_{exc,-1,n} = \sum_{j=1}^{N} \lambda_{nj} \tag{2.8}$$

where λ_{nj} is the number of optical channels between node n and node j.

The equation 2.9 refers to the number of ports in optical switch in node n, $P_{oxc,n}$, i.e. the number of line ports and the number of adding ports of node n which can be calculated as

$$P_{oxc,n} = \sum_{j=1}^{N} f_{nj}^{od} + \sum_{j=1}^{N} \lambda_{nj} \tag{2.9}$$

where f_{nj}^{od} refers to the number of line ports for all demand pairs (od) and λ_{nj} refers to the number of add ports.

To implement this heuristic approach there are used algorithms made in Java in a programming software called Eclipse and they are tested in an open-source network program called Net2Plan. In the Net2Plan guide section **??** there is an explanation on how to use and test them in this network planner.

In the next pages it will be described all the steps performed to obtain the final results in the translucent transport mode with 1+1 protection. In the figure below 2.79 it is shown a fluxogram with the description of this transport mode approach.

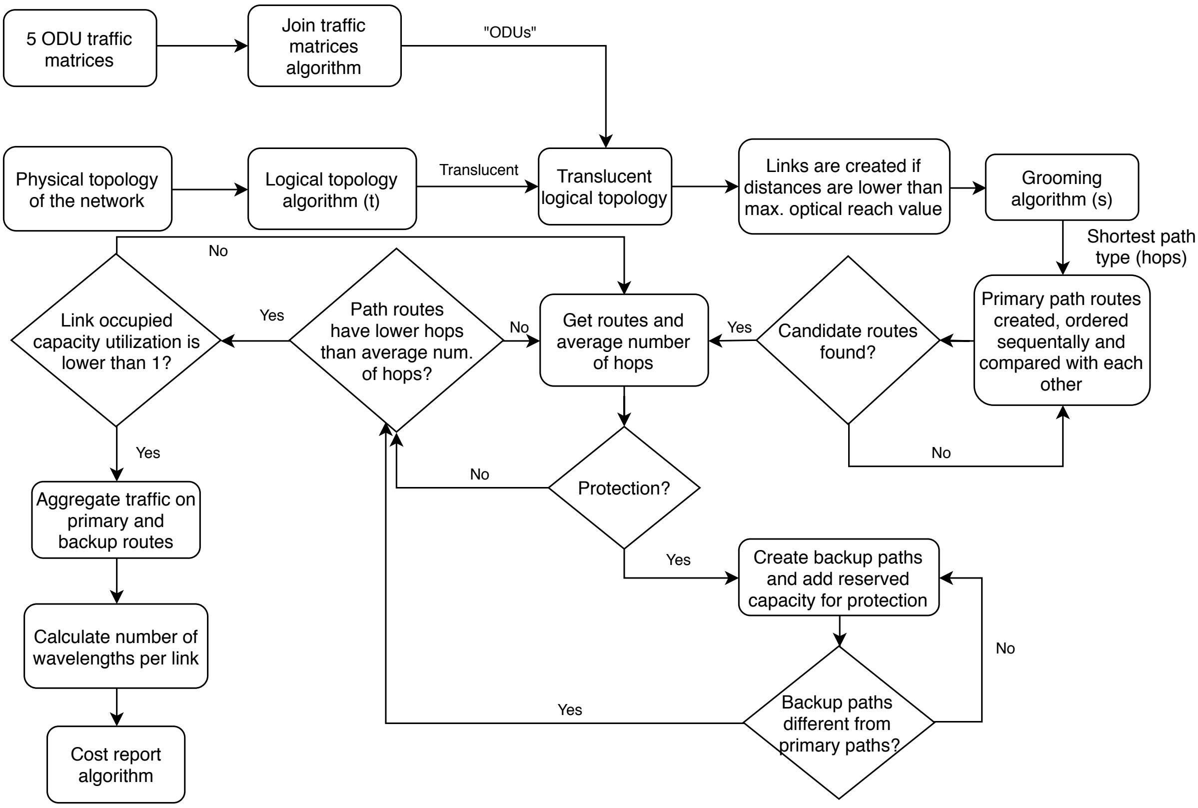

Figura 2.79: Fluxogram with the steps performed in the translucent with 1+1 protection transport mode approach.

Creation and join the traffic matrices

The first step is to create the traffic matrices based on the reference network **??**. In order to create the 5 traffic matrices in Net2Plan it is necessary the length of all the links and the total traffic used in this network, so later it is needed to define in Net2Plan the length in all end nodes and the total traffic depends on the value of traffic used (low traffic - 0.5 Tbit/s, medium traffic - 5 Tbit/s and high traffic - 10 Tbit/s). As you can see in the figure below, it is defined the path of the 5 ODUs and they will be aggregated in just one single ODU, making it possible to join all the demands in just one file and load it later into the network. This final resulting ODU joins the multiple traffic demands from all the traffic matrices previously created and, of course, the traffic demands will depend on the values used on the creation of the matrices (low, medium and high traffic).

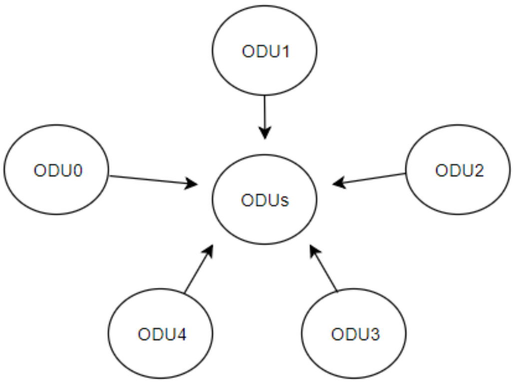

Figura 2.80: Join the 5 ODU traffic matrices into 1 single file "ODUs". The 5 traffic demands from the traffic matrices previously created are joined into 1 file to load it later on Net2Plan.

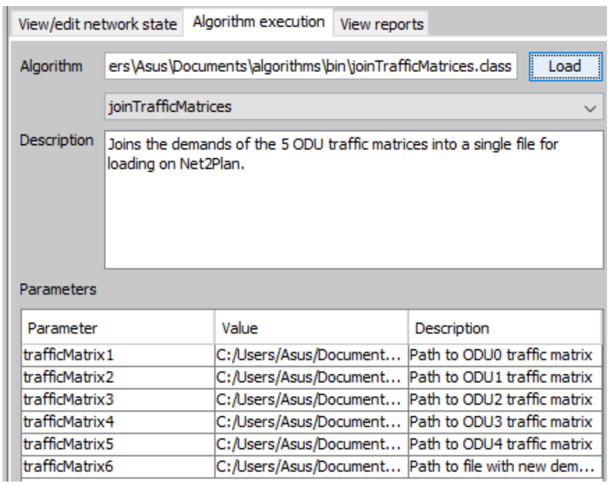

Figura 2.81: Load of the join traffic matrices algorithm for the translucent transport mode on Net2Plan. It is defined the 5 paths to load the 5 ODU traffic matrices and the last path is the one where will be saved the file that joins all 5 the traffic demands.

Creation of the physical topology

The next step is to create the allowed physical topology of the network in Net2Plan. This network consists in 6 nodes and 8 bidirectional links. It is now also possible to define the length in all links. In the figure below it is shown the allowed physical topology in this transport mode.

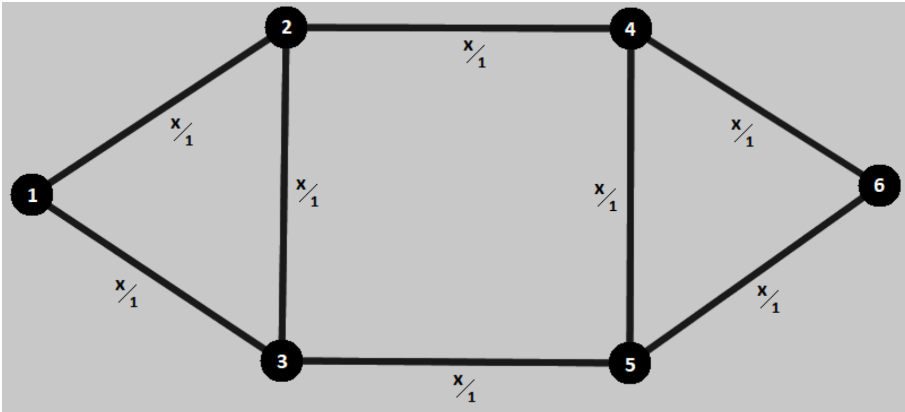

Figura 2.82: Allowed physical topology. The allowed physical topology is defined by the duct and sites in the field. It is assumed that each duct supports up to 1 bidirectional transmission system and each site supports up to 1 node.

Creation of the logical topology

It is now time to create the allowed logical topology. A network topology represents how the links and the nodes of the network interconnect with each other and the logical topology algorithm creates the logical topology on another layer. In the translucent transport mode each node connects to other node if the shortest path distance between them is lower than the maximum optical reach value (in km) that an optical signal can traverse in an optical channel. If the maximum optical reach is higher than the shortest path of a link, then it is not created a logical link of that specific physical link. On the contrary, there are created direct links between all node pairs in the network that follow this rule. These additions of links between end nodes are made in the new upper layer of the network. The respective demands are saved in the new upper layer and those demands from the lower layer are then removed. The lower layer is the physical layer of the network and it is now created a new upper layer which is the logical layer of the network and represents the logical topology of the translucent transport mode. The allowed physical and optical topologies, the logical topologies for all ODUs and the resulting physical topology is shown in the next section below 2.7.14 for the three traffic scenarios. It is shown below three figures with the code in Java of the creation of the network logical topology, the load of the logical topology algorithm in Net2Plan and the resulting allowed optical topology for the translucent transport mode with 1+1 protection.

```
if (netPlan.isSingleLayer() && logicalTopology.equalsIgnoreCase("Translucent")) {

    int maximumOpticalReach = Integer.parseInt(algorithmParameters.get("maximumOpticalReach"));
    maxOpticalReach = maximumOpticalReach;

    sendToFile("opticalReach.txt");

    this.lowerLayer = netPlan.getNetworkLayerDefault();
    lowerLayer.setName("Physical Topology");
    this.upperLayer = netPlan.addLayer("Logical Topology Translucent","Upper layer of the design","ODU","ODU",null);
    upperLayer.setDescription("Translucent Logical Topology"+" - Maximum Optical Reach= "+maximumOpticalReach+" km");
    netPlan.removeAllLinks(upperLayer);

    for (Node i : netPlan.getNodes()) {
        for (Node j : netPlan.getNodes()) {
            if (i.getIndex() != j.getIndex()) {
                if (netPlan.getNodePairEuclideanDistance(i, j) <= maximumOpticalReach) {
                    netPlan.addLink(i, j, 0, netPlan.getNodePairEuclideanDistance(i, j), 200000, null, upperLayer);
                }
            }
        }
    }
}
```

Figura 2.83: Java code of the logical topology approach for the translucent transport mode. The logical layer is created by adding direct links between all end nodes if their shortest path between them is lower than the maximum optical reach value. The new layer is now the translucent logical topology of the network.

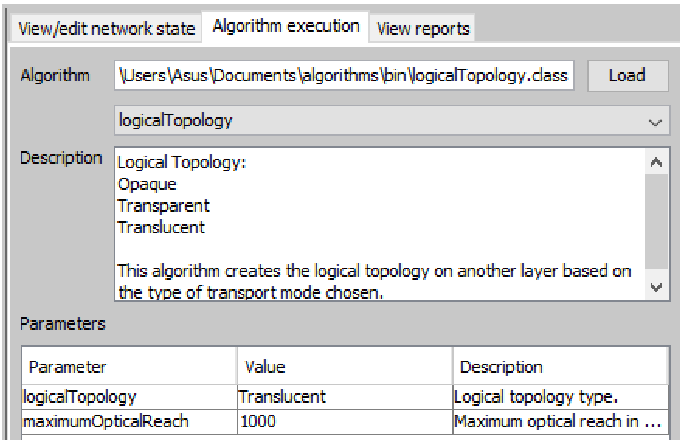

Figura 2.84: Load of the logical topology algorithm for the translucent transport mode on Net2Plan. It is assumed that the maximum optical reach value is 1000 km, so all direct links between node pairs with different index are created in all optical channels.

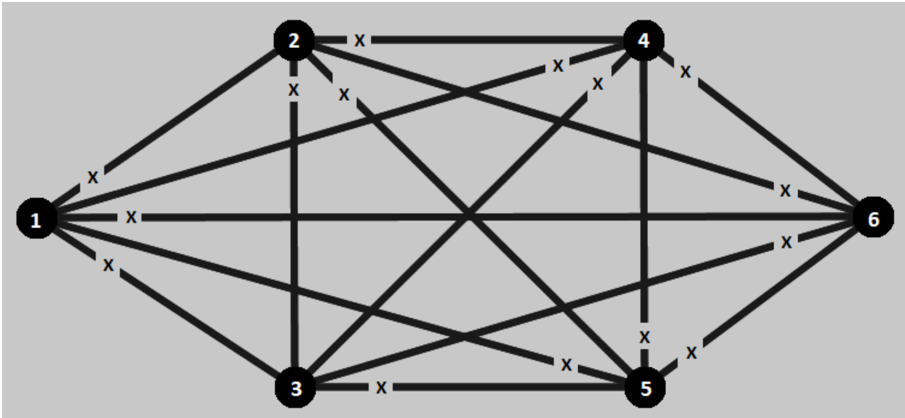

Figura 2.85: Allowed optical topology. It is assumed that each connections between demands supports up to 100 lightpaths. In the translucent transport mode there are several possible logical topologies for a specific network. It depends, for example, on the maximum optical reach value that represents the maximum distance that an optical signal can traverse a link without the need of regenerate.

Creation of routes and aggregation of traffic

After a network topology is created, it is now time to set the routing and then grooming algorithms. In the translucent with 1+1 protection transport mode the objective function is to minimize the lightpaths of the network and to find a route for every lightpath demand without considering the wavelength assignment. In this heuristic approach the goal is to minimize the lightpaths blocked and going through the carried lightpaths search for the ones that minimizes the average number of physical hops. Some of the links in the network will suffer from excessive traffic and some other links will occupy only a small part of the bandwidth. In order to do more efficiently, the link's utilizations are almost 1 so there is less non occupied capacity. Firstly, the candidate routes are found by the Dijkstra algorithm for each pair of nodes and calculates the minimum distance for each source and destination nodes and compares with each other. Then, the wavelength assignments are carried out for every lightpaths that have the minimum average of hops for each route and if the path distance is lower than the maximum optical reach value (in this report the maximum optical reach value used is 1000 km). The algorithm will compare the previous candidate routes that will be saved to a list with the new ones that will be created. If the new routes are different from the previously created ones, if they are the next shortest path routes and if they have the same restrictions as in the translucent without survivability transport mode, then the algorithm will add these routes to the network and they will be the protection segments (backup paths) of the network. The carried lightpaths resulted previously are sequentially processed and the offered traffic demands will be also set into these protection path routes. The routes are saved to a "Set" of routes and in each link of end nodes it is set the traffic demands into these routes that will integrate the whole network. The final resulting backup path routes are used to prevent network failures. Despite of the fact the network will be much more secure, the network CAPEX will increase more than the double, due to the creation of primary and backup paths.

```
case "Logical Topology Translucent":

    try {
        readFile();
    } catch (IOException e) {
        e.printStackTrace();
    }

    int hops = 0;
    Set<Route> nRoutes = new HashSet<Route>();

    for (Demand d : netPlan.getDemands(lowerLayer)) {

        nRoutes = d.getRoutes();
        for (Route c : nRoutes) {
            hops += c.getNumberOfHops();
        }
    }
    int n = hops/netPlan.getNumberOfRoutes(lowerLayer);

    for (Demand d : netPlan.getDemands(lowerLayer)) {
        boolean odd = true;
        int counter = 0;

        Set<Route> droutes = d.getRoutes();
        System.out.println(droutes.size());

        for (Route c : droutes) {
            counter++;
            boolean jump = false;
```

Figura 2.86: Creation of routes and aggregation of traffic for the translucent with 1+1 protection transport mode. The candidate routes are searched by the shortest path type method and the average minimum number of physical hops of the routes.

```
else {
  if (protection) {
      List<Link> workingpath = save.getSeqLinksRealPath();

      System.out.println("Protection-Translucent");
      for (Link t : workingpath) {
          if (c.getSeqLinksRealPath().contains(t)) {
              jump = true;
              break;
          }
      }

      if (jump == false) {
          if (c.getNumberOfHops() <= (n-1) && c.getLengthInKm() <= maxOpticalReach) {
              ProtectionSegment segment=netPlan.addProtectionSegment(c.getSeqLinksRealPath(),d.getOfferedTraffic(),null);
              save.addProtectionSegment(segment);
          }
          odd = true;
          break;
      }

      if (jump == true && counter == droutes.size()) {
          ProtectionSegment segment=netPlan.addProtectionSegment(c.getSeqLinksRealPath(),d.getOfferedTraffic(),null);
          save.addProtectionSegment(segment);
          odd = true;
          throw new Net2PlanException("Number of routes is not enough");
      }
  }
```

Figura 2.87: Creation of routes and aggregation of traffic for the translucent with 1+1 protection transport mode. The protection segments are added to all the primary paths that were chosen by the shortest path type method.

```
netPlan.removeAllRoutesUnused(1);

ArrayList<Long> tNodeIdsTransl = netPlan.getNodeIds();
Node inTransl;
Node outTransl;
Set<Route> groomRouteTransl;
Set<ProtectionSegment> protectRoutesTransl;
Route compareTransl = null;
Route compareTransl2 = null;
ProtectionSegment compare1Transl = null;
List<Link> pathTransl;
List<Link> pathTransl2;
int nWTransl = 0;

for (long tNodeId : tNodeIdsTransl) {
    inTransl = netPlan.getNodeFromId(tNodeId);
    for (long tNodeId1 : tNodeIdsTransl) {

        if (tNodeId == tNodeId1)
            continue;

        outTransl = netPlan.getNodeFromId(tNodeId1);
        double totaltraffic = 0;

        groomRouteTransl = netPlan.getNodePairRoutes(inTransl, outTransl, false, lowerLayer);
        protectRoutesTransl = netPlan.getNodePairProtectionSegments(inTransl, outTransl, false, lowerLayer);
```

Figura 2.88: Creation of routes and aggregation of traffic for the translucent with 1+1 protection transport mode. The traffic demands are set into the candidate primary path routes found earlier and also compared with the backup path routes.

```
// Protection Segments
totaltraffic = 0;

for (ProtectionSegment protect : protectRoutesTransl) {
    if(protect.getCarriedTraffic() <= protect.getCapacity()){
        totaltraffic = totaltraffic + protect.getReservedCapacityForProtection();
        compare1Transl = protect;
    }
}

if (protection) {
    pathTransl = compare1Transl.getSeqLinks();
}
```

Figura 2.89: Creation of routes and aggregation of traffic for the translucent with 1+1 protection transport mode. Minimizing the blocked traffic, the traffic demands are also set into the protection path routes and the wavelength assignment is carried out.

Function	Definition
netPlan.getDemands(lowerLayer)	Returns the array of demands for the lower layer.
d.getRoutes()	Returns all the routes associated to the demand "d".
c.getNumberOfHops()	Returns the route number of traversed links.
c.getLengthInKm()	Returns the route length in km, summing the traversed link lengths, as many times as the link is traversed.
c.setCarriedTraffic()	Sets the route carried traffic and the occupied capacity in the links, setting it up to be the same in all links.
d.getOfferedTraffic()	Returns the offered traffic of the demand "d".
save.getSeqLinksRealPath()	Returns the links of routes ordered sequentially.
save.addProtectionSegment(segment)	Add "segment" as a protection path in the route "save".
netPlan.getNodeIds()	Returns the array of the nodes' indexes.
netPlan.getNodeFromId(tNodeId)	Returns the node with the index "tNodeId".
netPlan.getNodePairRoutes (in,out,false,lowerLayer)	Returns the routes at "lowerLayer" from nodes "in" and "out".
netPlan.getNodePairProtectionSegments (in,out,false,lowerLayer)	Returns the protection segments at "lowerLayer" from nodes "in" and "out".
c.getCarriedTraffic()	Returns the route carried traffic at that moment.
protect.getCapacity()	Returns the link capacity.

Tabela 2.124: Table with the description of the main functions in the creation of routes and aggregation of traffic in the grooming algorithm.

Calculation of the number of wavelengths per link

The final step of the routing and grooming algorithms is to calculate the number of wavelengths per link for the whole network. This is the last and an important step because with the number of wavelengths per link in the network, it is possible to calculate other network components. In the translucent transport mode, as in the figure below shows, the algorithm starts with going through all the nodes which have different index between them (end nodes) and if the path distance between them is lower than the maximum optical reach value and in all the links that crosses between these pairs of nodes is reserved a link capacity based on the previous traffic aggregation on routes. The total carried traffic in the link including protection and non-protection segments will be divided by the wavelength capacity and it is now possible to obtain the number of wavelengths per link.

```java
for (Link link : pathTransl) {
    String nw = link.getAttribute("nW");
    if (nw != null) {
        nWTransl = Integer.parseInt(nw);
        nWTransl = (int) (nWTransl + Math.ceil(totaltraffic / wavelengthCapacity));
        link.setAttribute("nW", String.valueOf(nWTransl));
    } else {
        nWTransl = (int) Math.ceil(totaltraffic / wavelengthCapacity);
        link.setAttribute("nW", String.valueOf(nWTransl));
    }
}
```

Figura 2.90: Calculation of the number of wavelengths per link for the translucent transport mode. The link capacity is reserved based on the previous traffic aggregation.

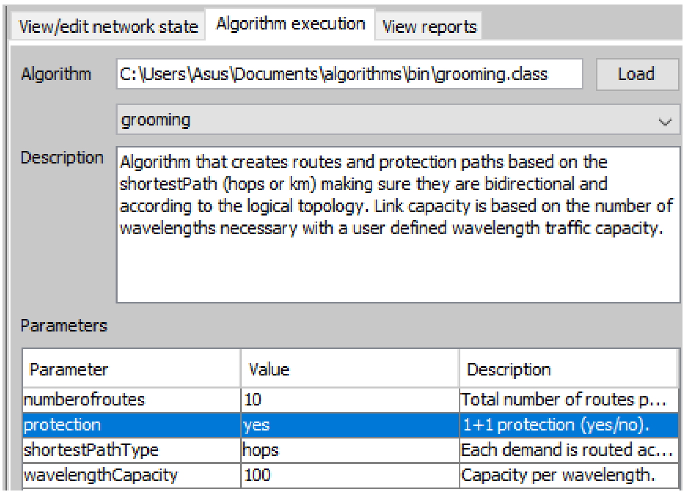

Figura 2.91: Load of the grooming algorithm for the translucent with 1+1 protection transport mode. The total number of routes per demand is set to 10, the user can define if the model is with or without protection, the shortest path type is set to "hops" and the capacity per wavelength is used 100 optical channels.

Network cost report

In order to obtain the network CAPEX results, the formulas needed to calculate the network elements and that are demonstrated previously in the beginning of this section 2.7.14 were "translated" into Java code in a cost report algorithm. This algorithm can be loaded in Net2Plan and calculates and shows in tables the network CAPEX and also the per-link and per-node information with more details.

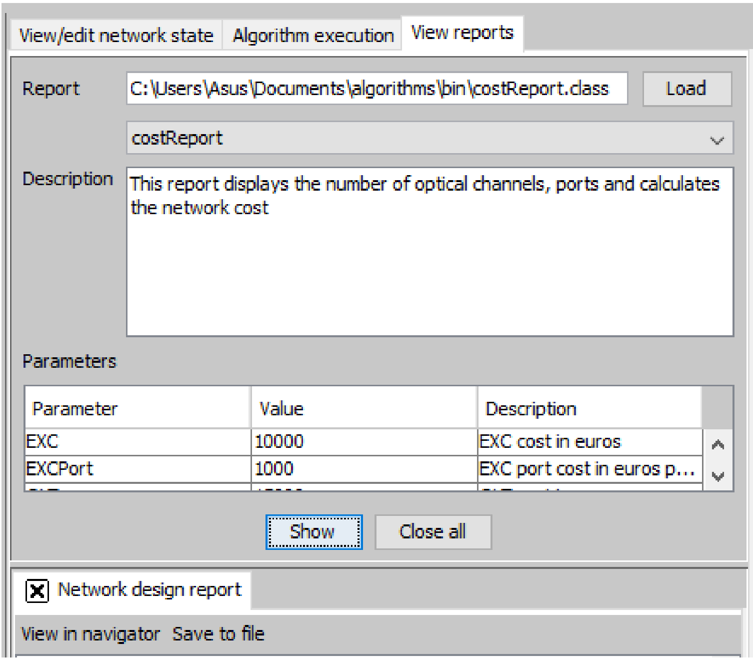

Figura 2.92: Load of the cost report algorithm on Net2Plan. The result view is an HTML page with the network optical and electrical components and their costs.

Result description

It is already known all the necessary formulas to obtain the CAPEX value for the reference network **??**. As described in the subsection of the network traffic **??**, it is necessary to obtain three different values of CAPEX for the low (0.5 Tbit/s), medium (5 Tbit/s) and high (10 Tbit/s) traffic. It is used a network software program called Net2Plan which can design the traffic matrices, create all the network topologies, simulate the algorithms into the network implemented in the programming software called Eclipse and analyze the results obtained. In this chapter will be demonstrated the results by Pedro's heuristics. In each of the three traffic scenarios, it will be shown the network topologies followed by the table with the CAPEX value of the network.

Low Traffic Scenario:

In this scenario we have to take into account the traffic calculated in **??**. In a first phase we will show the various existing topologies of the network. The first are the allowed topologies, physical and optical topologies, the second are the logical topology for all ODUs and finally the resulting physical topology.

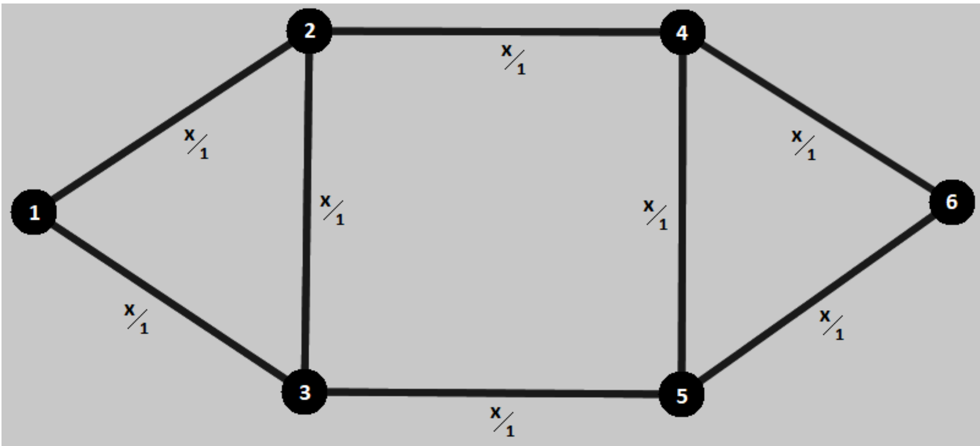

Figura 2.93: Allowed physical topology. The allowed physical topology is defined by the duct and sites in the field. It is assumed that each duct supports up to 1 bidirectional transmission system and each site supports up to 1 node.

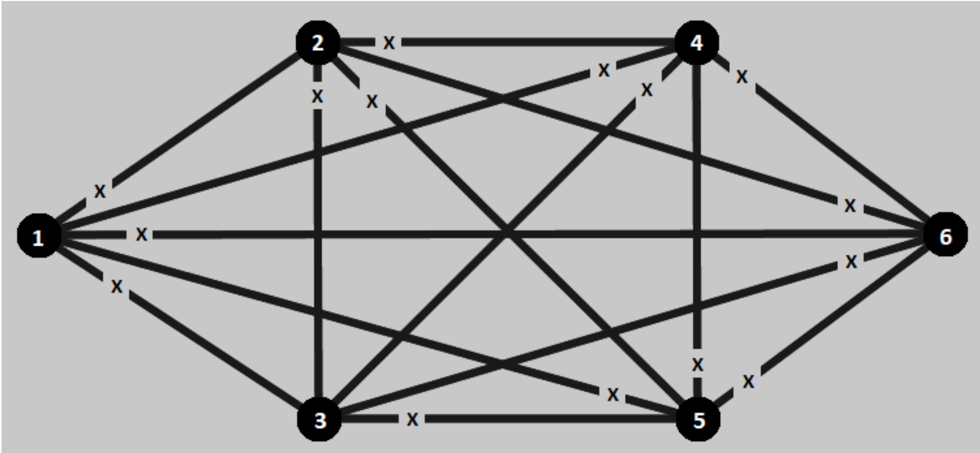

Figura 2.94: Allowed optical topology. The allowed optical topology is defined by the transport mode (translucent transport mode in this case). It is assumed that each connections between demands supports up to 100 lightpaths.

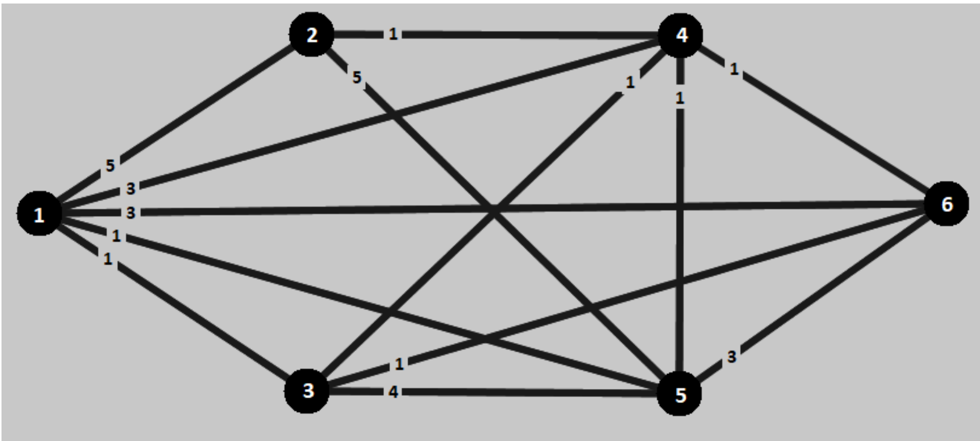

Figura 2.95: ODU0 logical topology defined by the ODU0 traffic matrix.

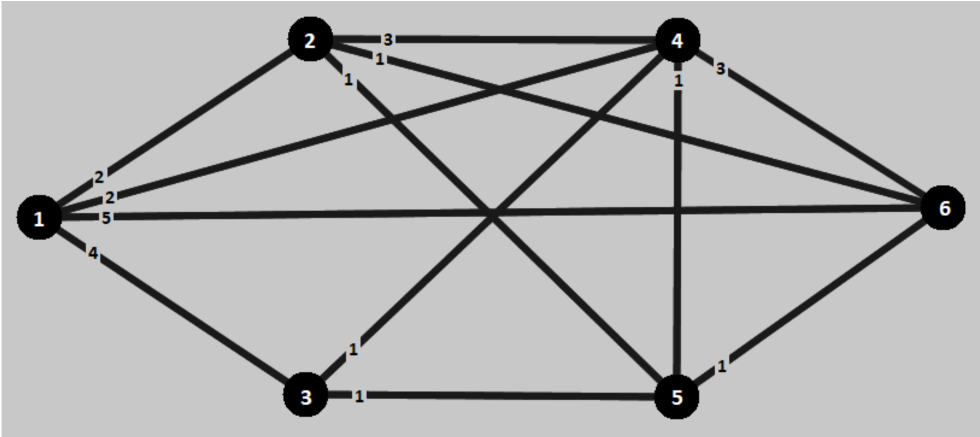

Figura 2.96: ODU1 logical topology defined by the ODU1 traffic matrix.

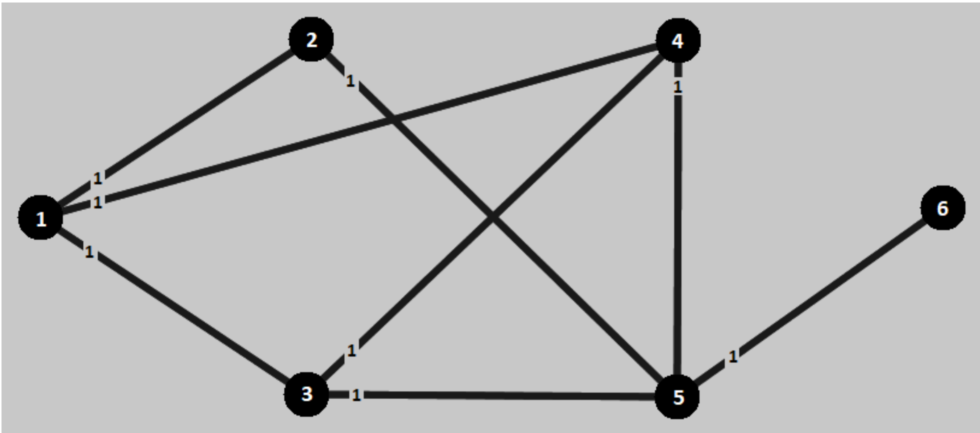

Figura 2.97: ODU2 logical topology defined by the ODU2 traffic matrix.

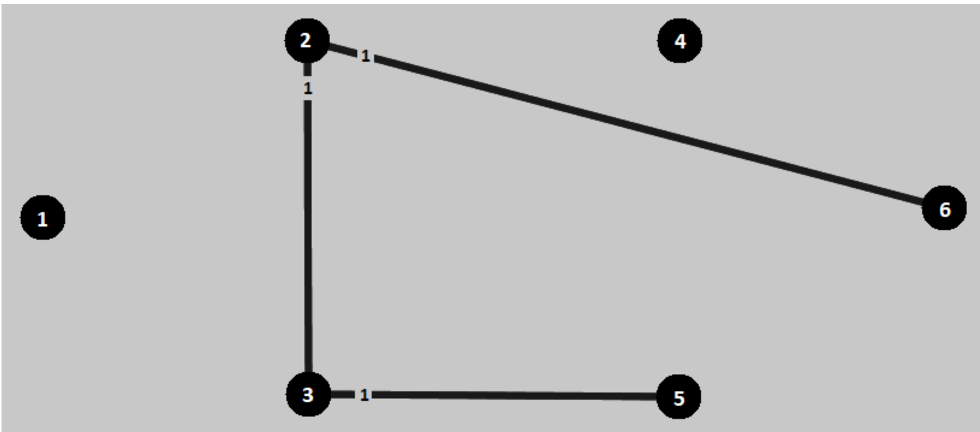

Figura 2.98: ODU3 logical topology defined by the ODU3 traffic matrix.

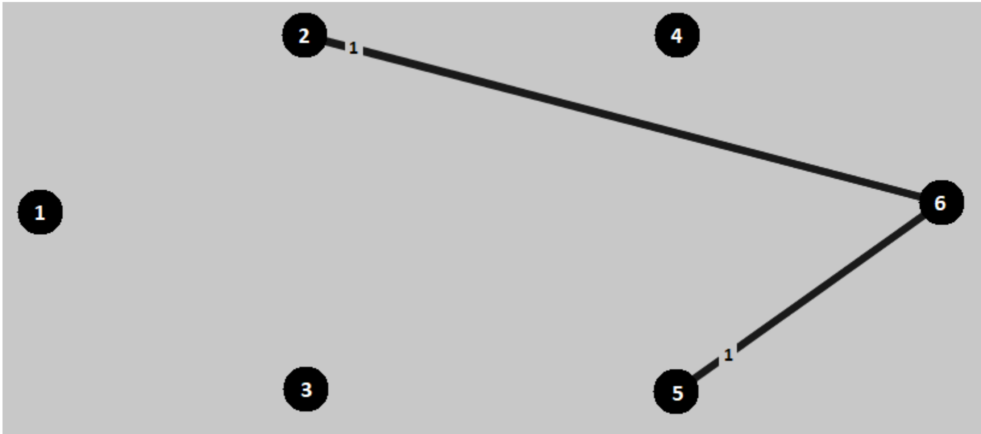

Figura 2.99: ODU4 logical topology defined by the ODU4 traffic matrix.

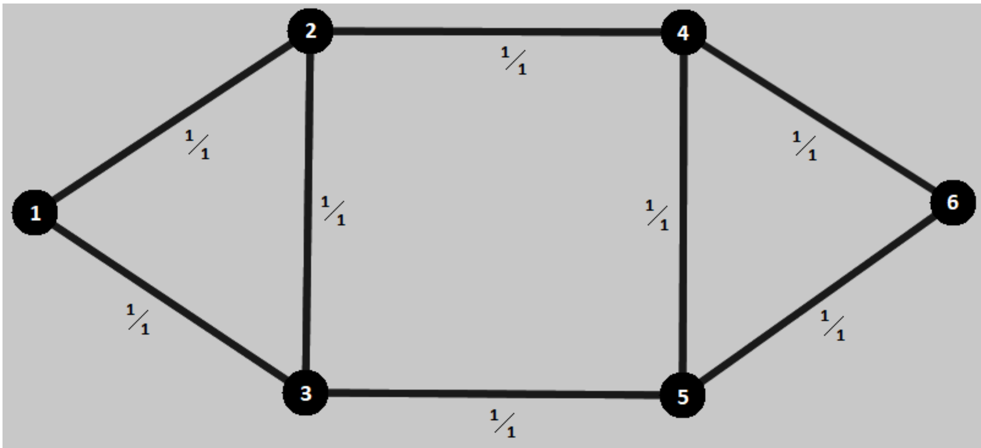

Figura 2.100: Physical topology after dimensioning.

Following all the steps mentioned in the **??**, applying the routing and grooming heuristic algorithms in the Net2Plan software and using all the data referring to this scenario, the obtained result for the Pedro's heuristics can be consulted in the following table 2.125. In table 2.120 mentioned in previous model we can see how all the values were calculated.

CAPEX of the Network						
			Quantity	Unit Price	Cost	Total
Link Cost	OLTs		16	15 000 €	240 000 €	23 520 000 €
	100 Gbits/s Transceivers		46	5 000 €/Gbit/s	23 000 000 €	
	Amplifiers		70	4 000 €	280 000 €	
Node Cost	Electrical	EXCs	6	10 000 €	60 000 €	2 142 590 €
		ODU0 Ports	60	10 €/port	600 €	
		ODU1 Ports	50	15 €/port	750 €	
		ODU2 Ports	16	30 €/port	480 €	
		ODU3 Ports	6	60 €/port	360 €	
		ODU4 Ports	4	100 €/port	400 €	
		Transponders	18	100 000 €/port	1 800 000 €	
	Optical	OXCs	6	20 000 €	120 000 €	
		Line Ports	46	2 500 €/port	115 000 €	
		Add Ports	18	2 500 €/port	45 000 €	
Total Network Cost						25 662 590 €

Tabela 2.125: Table with detailed description of CAPEX.

Medium Traffic Scenario:

In this scenario we have to take into account the traffic calculated in **??**. In a first phase we will show the various existing topologies of the network. The first are the allowed topologies, physical and optical topologies, the second are the logical topology for all ODUs and finally the resulting physical topology.

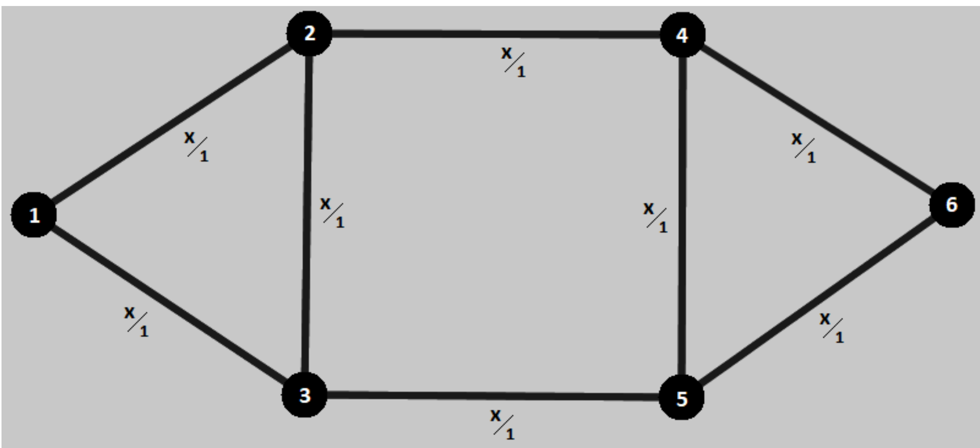

Figura 2.101: Allowed physical topology. The allowed physical topology is defined by the duct and sites in the field. It is assumed that each duct supports up to 1 bidirectional transmission system and each site supports up to 1 node.

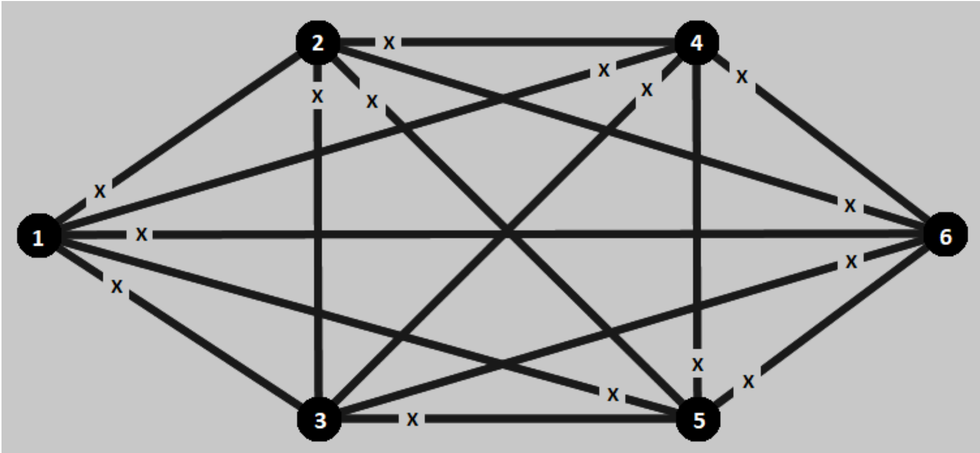

Figura 2.102: Allowed optical topology. The allowed optical topology is defined by the transport mode (translucent transport mode in this case). It is assumed that each connections between demands supports up to 100 lightpaths.

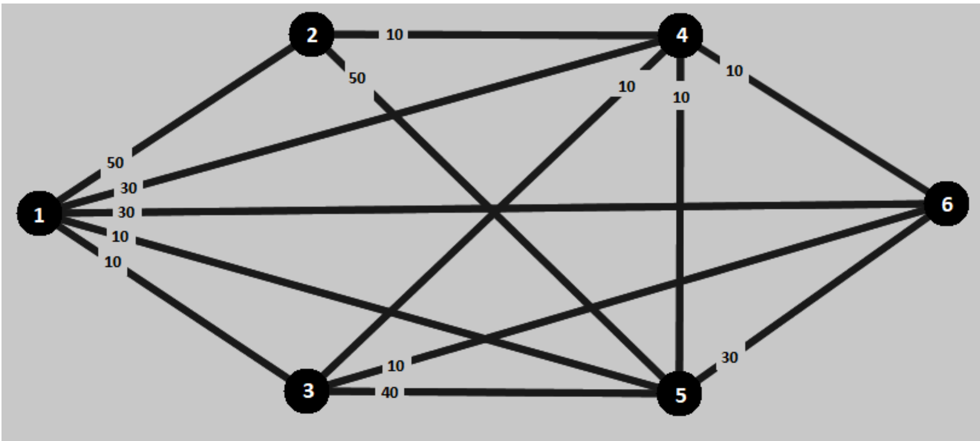

Figura 2.103: ODU0 logical topology defined by the ODU0 traffic matrix.

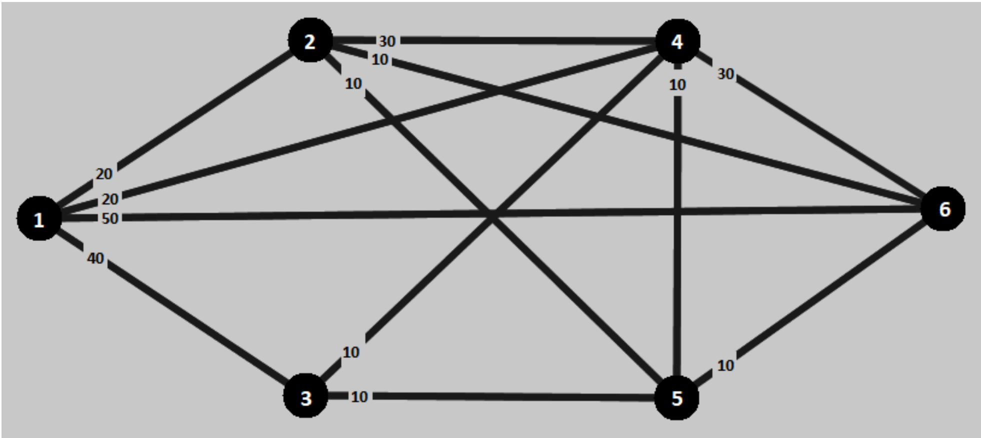

Figura 2.104: ODU1 logical topology defined by the ODU1 traffic matrix.

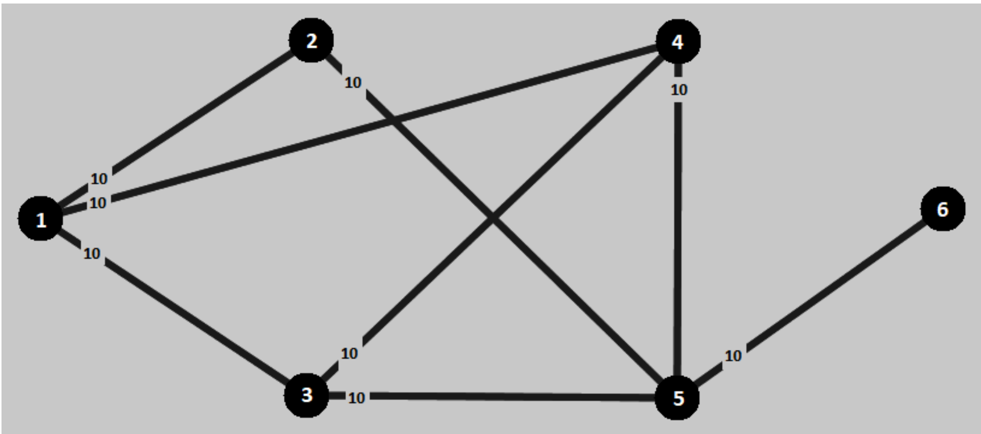

Figura 2.105: ODU2 logical topology defined by the ODU2 traffic matrix.

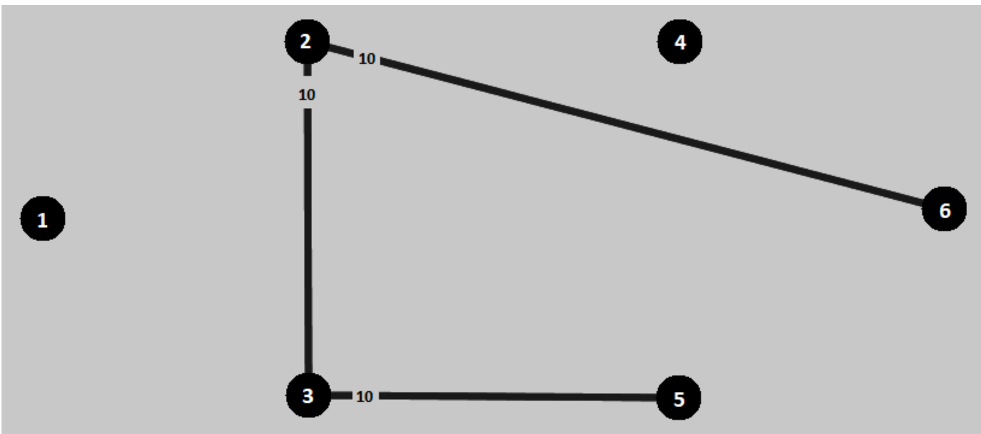

Figura 2.106: ODU3 logical topology defined by the ODU3 traffic matrix.

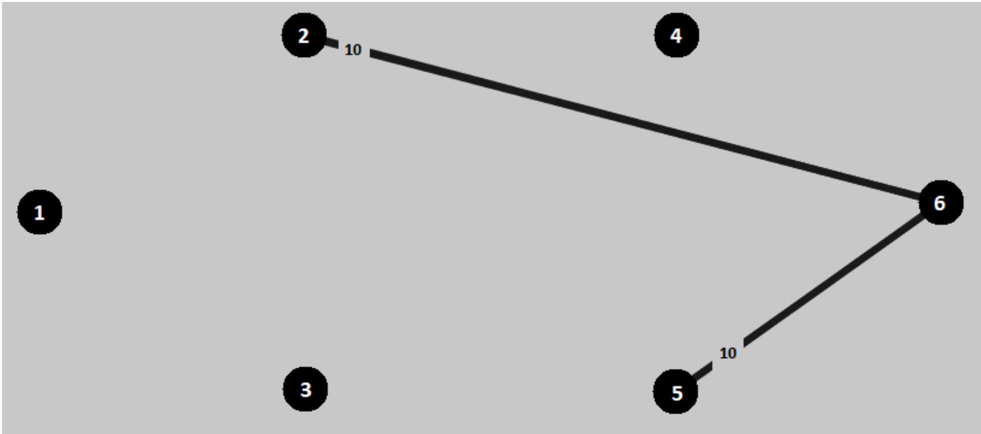

Figura 2.107: ODU4 logical topology defined by the ODU4 traffic matrix.

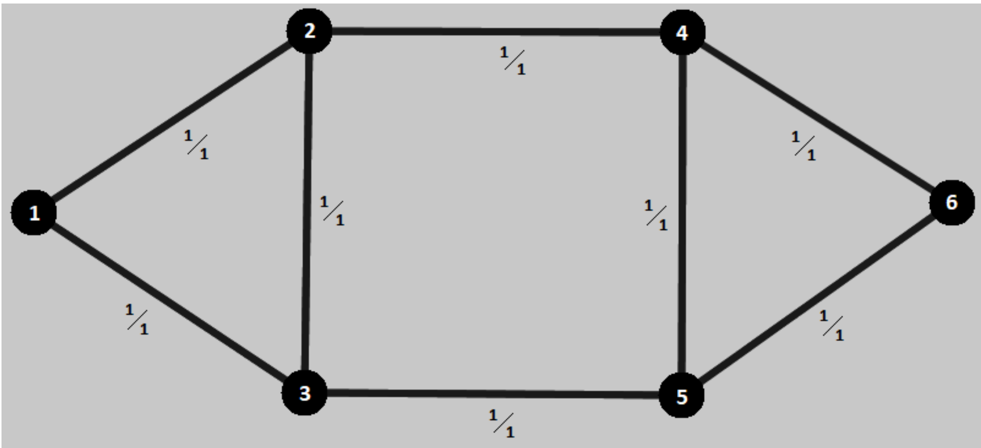

Figura 2.108: Physical topology after dimensioning.

Following all the steps mentioned in the **??**, applying the routing and grooming heuristic algorithms in the Net2Plan software and using all the data referring to this scenario, the obtained result for the Pedro's heuristics can be consulted in the following table 2.126. In table 2.120 mentioned in previous model we can see how all the values were calculated.

CAPEX of the Network						
			Quantity	Unit Price	Cost	Total
Link Cost	OLTs		16	15 000 €	240 000 €	78 520 000 €
	100 Gbits/s Transceivers		156	5 000 €/Gbit/s	78 000 000 €	
	Amplifiers		70	4 000 €	280 000 €	
Node Cost	Electrical	EXCs	6	10 000 €	60 000 €	8 795 900 €
		ODU0 Ports	600	10 €/port	6 000 €	
		ODU1 Ports	500	15 €/port	7 500 €	
		ODU2 Ports	160	30 €/port	4 800 €	
		ODU3 Ports	60	60 €/port	3 600 €	
		ODU4 Ports	40	100 €/port	4 000 €	
		Transponders	80	100 000 €/port	8 000 000 €	
	Optical	OXCs	6	20 000 €	120 000 €	
		Line Ports	156	2 500 €/port	390 000 €	
		Add Ports	80	2 500 €/port	200 000 €	
Total Network Cost						87 315 900 €

Tabela 2.126: Table with detailed description of CAPEX.

High Traffic Scenario:

In this scenario we have to take into account the traffic calculated in **??**. In a first phase we will show the various existing topologies of the network. The first are the allowed topologies, physical and optical topologies, the second are the logical topology for all ODUs and finally the resulting physical topology.

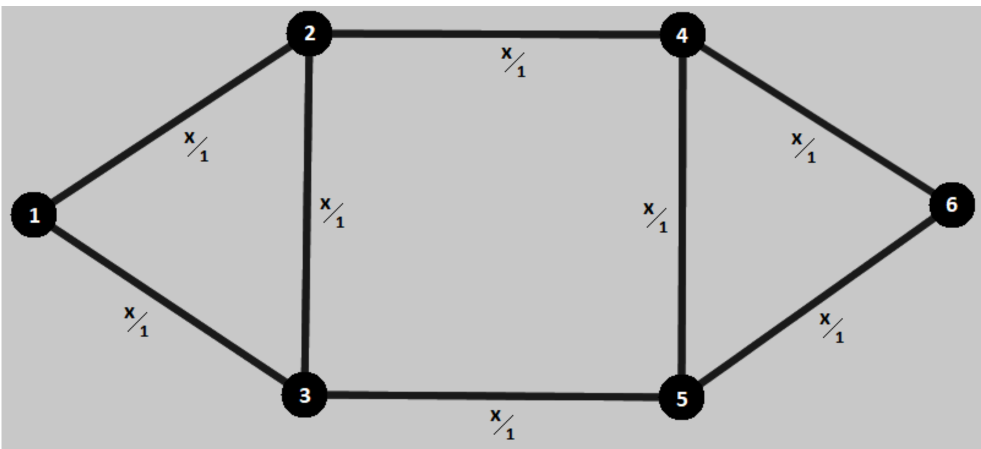

Figura 2.109: Allowed physical topology. The allowed physical topology is defined by the duct and sites in the field. It is assumed that each duct supports up to 1 bidirectional transmission system and each site supports up to 1 node.

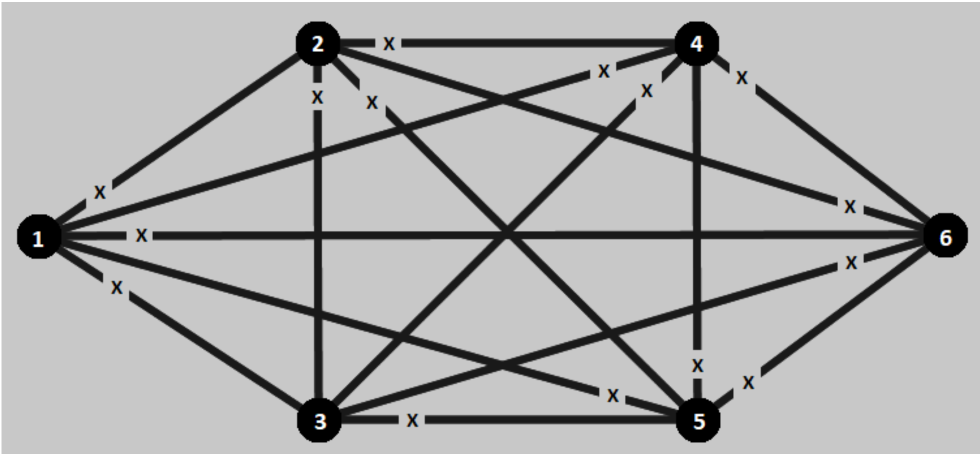

Figura 2.110: Allowed optical topology. The allowed optical topology is defined by the transport mode (translucent transport mode in this case). It is assumed that each connections between demands supports up to 100 lightpaths.

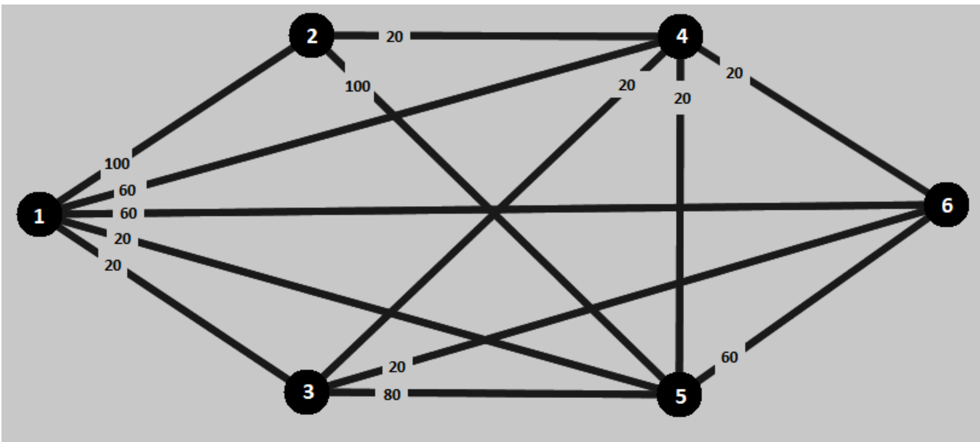

Figura 2.111: ODU0 logical topology defined by the ODU0 traffic matrix.

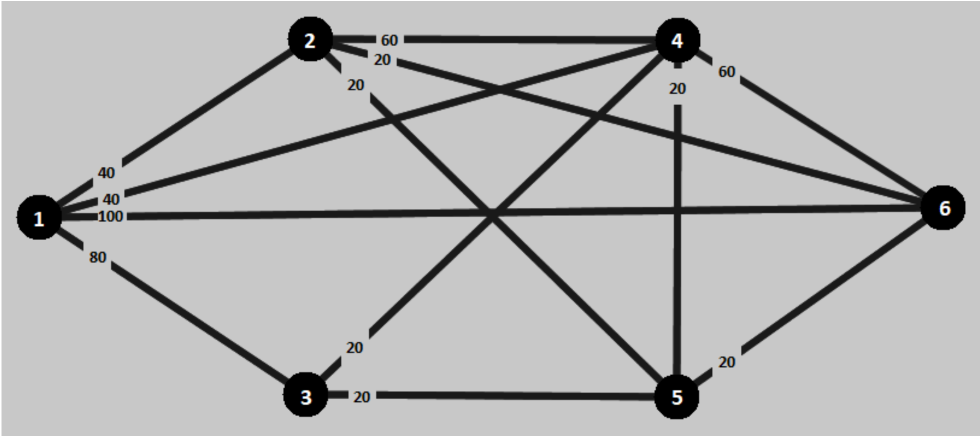

Figura 2.112: ODU1 logical topology defined by the ODU1 traffic matrix.

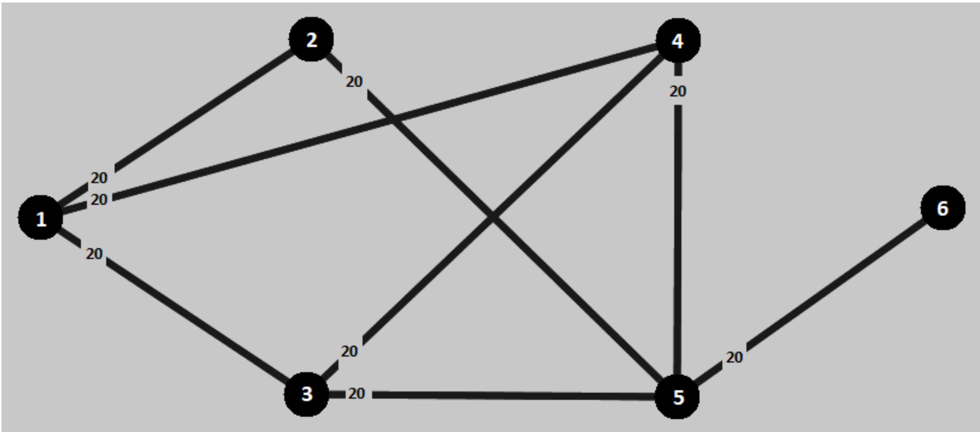

Figura 2.113: ODU2 logical topology defined by the ODU2 traffic matrix.

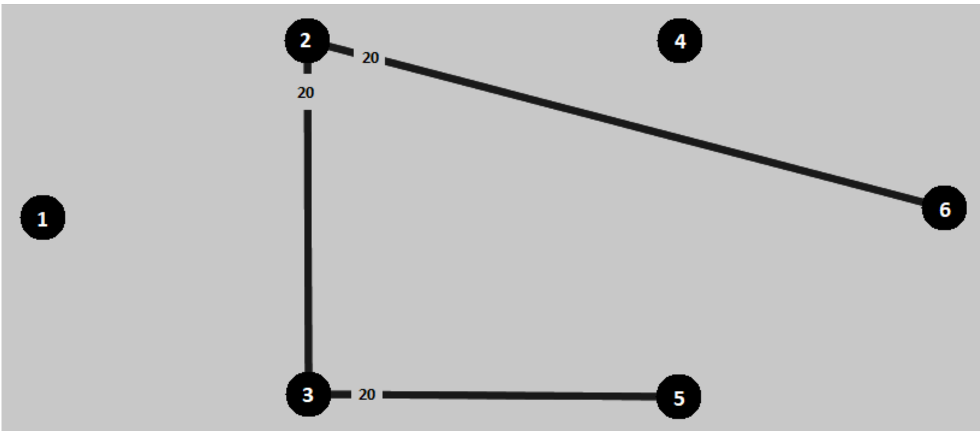

Figura 2.114: ODU3 logical topology defined by the ODU3 traffic matrix.

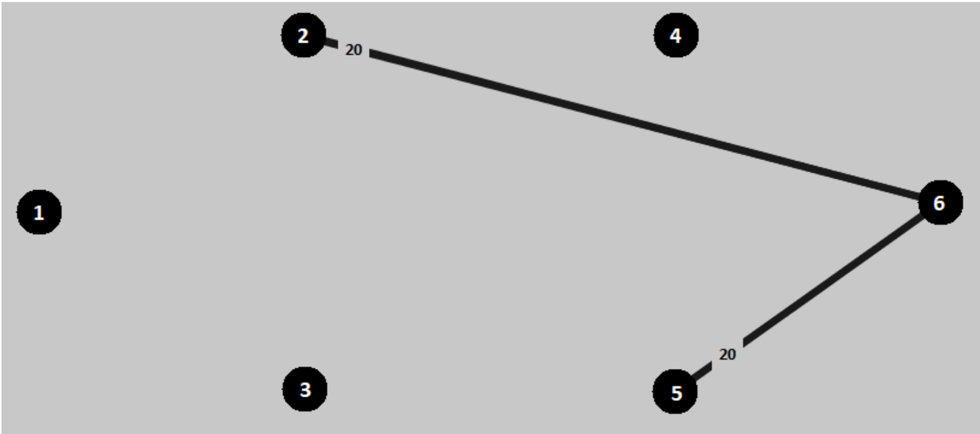

Figura 2.115: ODU4 logical topology defined by the ODU4 traffic matrix.

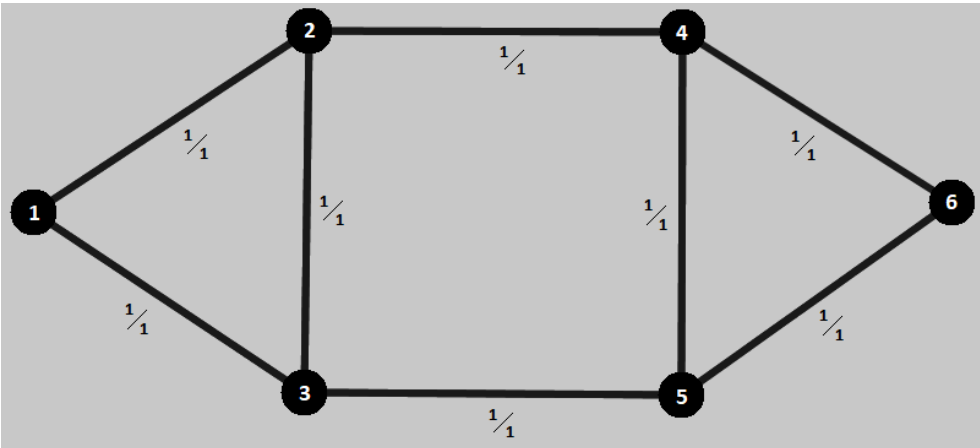

Figura 2.116: Physical topology after dimensioning.

Following all the steps mentioned in the **??**, applying the routing and grooming heuristic algorithms in the Net2Plan software and using all the data referring to this scenario, the obtained result for the Pedro's heuristics can be consulted in the following table 2.122. In table 2.120 mentioned in previous model we can see how all the values were calculated.

<table>
<tr><th colspan="7">CAPEX of the Network</th></tr>
<tr><th colspan="3"></th><th>Quantity</th><th>Unit Price</th><th>Cost</th><th>Total</th></tr>
<tr><td rowspan="3">Link Cost</td><td colspan="2">OLTs</td><td>16</td><td>15 000 €</td><td>240 000 €</td><td rowspan="3">147 520 000 €</td></tr>
<tr><td colspan="2">100 Gbits/s Transceivers</td><td>294</td><td>5 000 €/Gbit/s</td><td>147 000 000 €</td></tr>
<tr><td colspan="2">Amplifiers</td><td>70</td><td>4 000 €</td><td>280 000 €</td></tr>
<tr><td rowspan="10">Node Cost</td><td rowspan="7">Electrical</td><td>EXCs</td><td>6</td><td>10 000 €</td><td>60 000 €</td><td rowspan="10">16 751 800 €</td></tr>
<tr><td>ODU0 Ports</td><td>1 200</td><td>10 €/port</td><td>12 000 €</td></tr>
<tr><td>ODU1 Ports</td><td>1 000</td><td>15 €/port</td><td>15 000 €</td></tr>
<tr><td>ODU2 Ports</td><td>320</td><td>30 €/port</td><td>9 600 €</td></tr>
<tr><td>ODU3 Ports</td><td>120</td><td>60 €/port</td><td>7 200 €</td></tr>
<tr><td>ODU4 Ports</td><td>80</td><td>100 €/port</td><td>8 000 €</td></tr>
<tr><td>Transponders</td><td>154</td><td>100 000 €/port</td><td>15 400 000 €</td></tr>
<tr><td rowspan="3">Optical</td><td>OXCs</td><td>6</td><td>20 000 €</td><td>120 000 €</td></tr>
<tr><td>Line Ports</td><td>294</td><td>2 500 €/port</td><td>735 000 €</td></tr>
<tr><td>Add Ports</td><td>154</td><td>2 500 €/port</td><td>385 000 €</td></tr>
<tr><td colspan="6">Total Network Cost</td><td>164 271 800 €</td></tr>
</table>

Tabela 2.127: Table with detailed description of CAPEX.

Conclusions

Once we have obtained the results for all scenarios for the translucent without survivability and translucent with 1+1 protection we will now draw some conclusions about these results. For a better analysis of the results will be created the table 2.128 with the number of line ports, tributary ports and transceivers because they are important values for the cost of CAPEX, the cost of links, the cost of nodes and finally the cost of CAPEX.

	Low Traffic	Medium Traffic	High Traffic
CAPEX without survivability	11 592 590 €	49 125 900 €	93 921 800 €
CAPEX/Gbit/s without survivability	23 185 €/Gbit/s	9 825 €/Gbit/s	9 392 €/Gbit/s
Traffic (Gbit/s)	500	5 000	10 000
Bidirectional Links used	8	8	8
Number of Add ports	18	80	154
Number of Line ports	46	156	294
Number of Tributary ports	136	1 360	2 720
Number of Transceivers	46	156	294
Link Cost	23 520 000 €	78 520 000 €	147 520 000 €
Node Cost	2 142 590 €	8 795 900 €	16 751 800 €
CAPEX	**25 662 590 €**	**87 315 900 €**	**164 271 800 €**
CAPEX/Gbit/s	**51 325 €/Gbit/s**	**17 463 €/Gbit/s**	**16 427 €/Gbit/s**

Tabela 2.128: Table with different value of CAPEX for this case.

Looking at the previous table we can make some comparisons between the translucent with 1+1 protection scenario:

- Comparing the low traffic with the others we can see that despite having an increase of factor ten (medium traffic) and factor twenty (high traffic), the same increase does not occur in the final cost (it is lower);

  This happens because the number of the transceivers is lower than expected which leads by carrying the traffic with less network components and, consequently, the network CAPEX is lower;

- Comparing the medium traffic with the high traffic we can see that the increase of the factor is double and in the final cost this factor is very close but still inferior;

  This happens because the number of the transceivers is also lower but very close to the expected;

- Comparing the CAPEX cost per bit we can see that in the low traffic the cost is higher than the medium and high traffic, which in these two cases the value is similar, but still inferior in the higher traffic. The difference between the low traffic and medium/high traffics is significantly high;

  This happens because the lower the traffic, the higher CAPEX/bit will be. We can see that in medium and high traffic the results tend to be one closer and lower value.

We can also make some comparisons between the translucent without survivability and translucent with 1+1 protection scenarios:

- We can see that in the translucent with 1+1 protection transport mode the CAPEX cost for all the three traffic is more than the double;

  This happens because in the translucent with 1+1 protection transport mode there is a need of having a primary and a backup path, in case of a network failure, and the backup path is typically longer;

- Comparing the CAPEX cost per bit we can see that has a similar case in both of the two scenarios. In the low traffic the cost is higher than the medium and high traffic, which in these two cases the value is similar;

  This happens because the lower the traffic, the higher CAPEX/bit will be. We can see that in medium and high traffic the results tend to be one closer and lower value.

Opens Issues

The creation of this model for any scenario, started with some considerations and some open issues being:

- Allow maximum optical reach value in optical channels.

  It is assumed that the maximum optical reach value is 1000 km, so all direct links between node pairs with different index are created in all optical channels;

- Allow multiple transmission system.

  The presented model for each link only supports one transmission system.