Traffic Engineering of Unicast Services

Modelação e Desempenho de Redes e Serviços

Prof. Amaro de Sousa (asou@ua.pt)

DETI-UA, 2022/2023

Traffic engineering of unicast services

A unicast service is defined by a set of point-to-point traffic flows on a given telecommunication network.

- Consider a network composed by a set of point-to-point links and supporting one unicast service defined by a set of traffic flows T, such that the packets of all flows have the same statistics.
  - The network is modelled by a graph $G=(N,A)$. Set N is the set of network nodes. Set A is the set of network links: the arc $(i,j) \in A$ represents the link between nodes $i \in N$ and $j \in N$ from i to j whose capacity is given by c_{ij} in bps (usually $c_{ij} = c_{ji}$).
  - Each traffic flow $t \in T$ is defined by its origin node o_t, destination node d_t, average throughput from origin to destination b_t (in bps) and average throughput from destination to origin $\underline{b}_t$ (in bps).
  - For each flow $t \in T$, P_t is the set of the candidate routing paths in graph G from its origin node o_t to its destination node d_t.

The <u>traffic engineering</u> task is the task of choosing for each flow $t \in T$ the percentage of its average throughput that must be routed through each of its candidate routing paths of P_t in each direction.

Traffic engineering with single path routing

- In single path routing, each traffic flow must be routed through one single path (no flow bifurcation is allowed).
- Symmetrical routing might be required or not; when required, the routing path from a node $j \in N$ to a node $i \in N$ must use the same links as the routing path from node $i \in N$ to node $j \in N$.

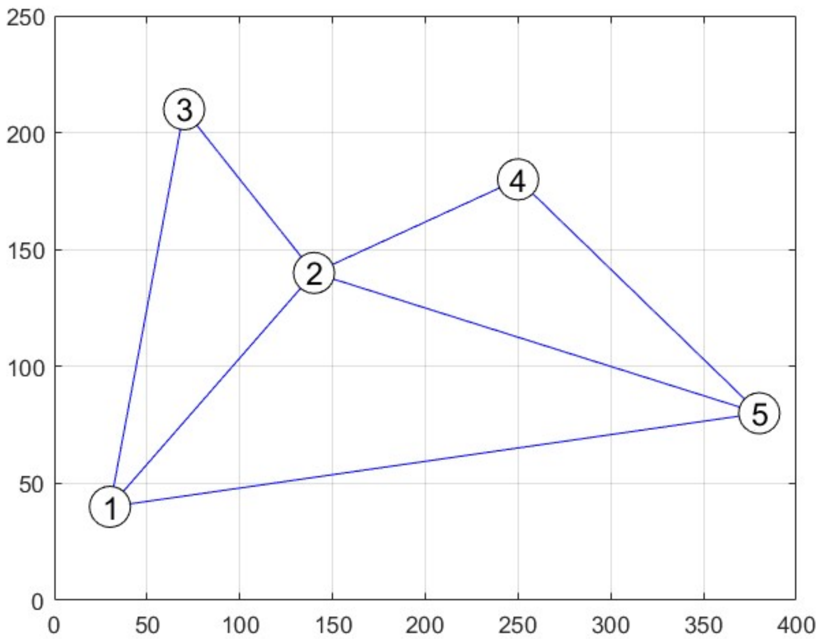
Consider a binary variable x_{tp} associated to each traffic flow $t \in T$ and each routing path $p \in P_t$ that, when is 1, indicates that traffic flow t is routed through path p.

Any traffic engineering solution with single path routing must be compliant with the following constraints:

- For each flow $t \in T$, one of its associated variables x_{tp} must be 1 and all other associated variables must be 0.
- At each arc $(i,j) \in A$, the sum of the throughput values (either b_t or $\underline{b}_t$) of all flows routed through it cannot be higher than its capacity c_{ij}.

Example

Example network:



All links with 10 Gbps of capacity
(in general, these values can be different)

Traffic flows:

t	o_t	d_t	b_t (Gbps)	$\underline{b}_t$ (Gbps)
1	1	4	1.0	1.0
2	1	5	2.0	2.0
3	2	4	3.0	3.0
4	3	5	2.0	2.0

In general, these values are different

Flow 1:
Path 1 = 1 2 4
Path 2 = 1 3 2 4
Path 3 = 1 5 4
Path 4 = 1 2 5 4
Path 5 = 1 3 2 5 4
Path 6 = 1 5 2 4

Flow 2:
Path 1 = 1 5
Path 2 = 1 2 5
Path 3 = 1 2 4 5
Path 4 = 1 3 2 5
Path 5 = 1 3 2 4 5

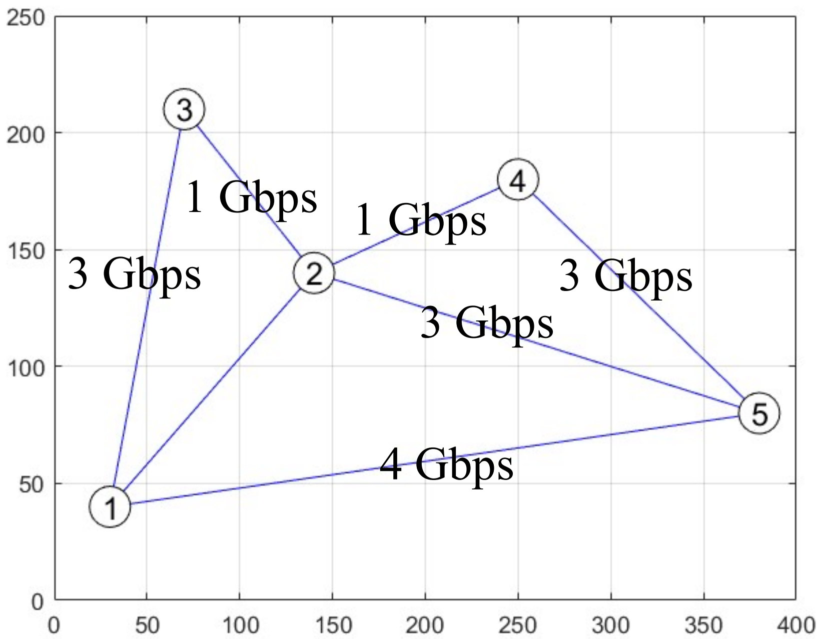
Flow 3:
Path 1 = 2 4
Path 2 = 2 5 4
Path 3 = 2 1 5 4
Path 4 = 2 3 1 5 4

Flow 4:
Path 1 = 3 2 5
Path 2 = 3 2 4 5
Path 3 = 3 1 5
Path 4 = 3 1 2 5
Path 5 = 3 2 1 5
Path 6 = 3 1 2 4 5

Routing paths ordered from shortest to longest lengths

Example: one possible solution

Example network:



All links with 10 Gbps of capacity

Traffic flows:

t	o_t	d_t	b_t (Gbps)	$\underline{b}_t$ (Gbps)
1	1	4	1.0	1.0
2	1	5	2.0	2.0
3	2	4	3.0	3.0
4	3	5	2.0	2.0

Flow 1:
- Path 1 = 1 2 4
- **Path 2 = 1 3 2 4**
- Path 3 = 1 5 4
- Path 4 = 1 2 5 4
- Path 5 = 1 3 2 5 4
- Path 6 = 1 5 2 4

Flow 2:
- **Path 1 = 1 5**
- Path 2 = 1 2 5
- Path 3 = 1 2 4 5
- Path 4 = 1 3 2 5
- Path 5 = 1 3 2 4 5

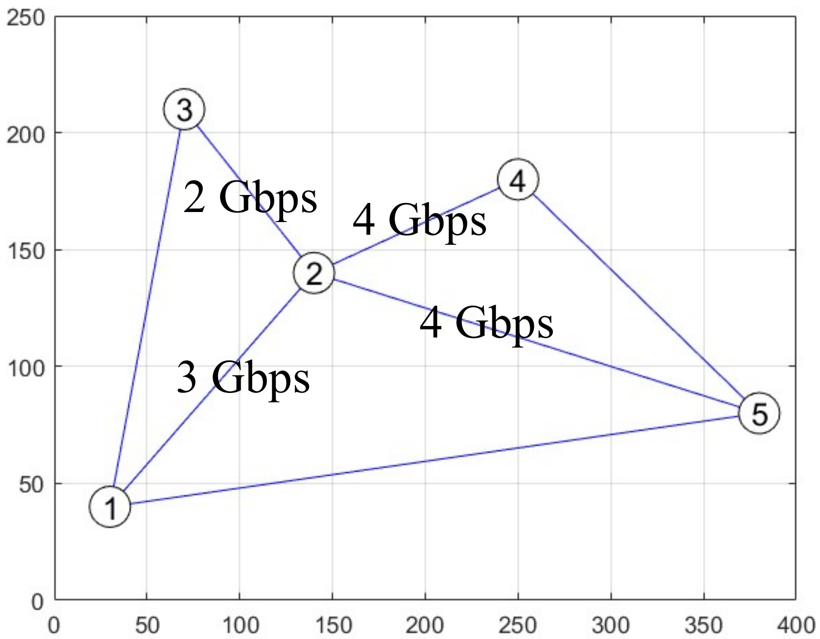
Flow 3:
- Path 1 = 2 4
- **Path 2 = 2 5 4**
- Path 3 = 2 1 5 4
- Path 4 = 2 3 1 5 4

Flow 4:
- Path 1 = 3 2 5
- Path 2 = 3 2 4 5
- **Path 3 = 3 1 5**
- Path 4 = 3 1 2 5
- Path 5 = 3 2 1 5
- Path 6 = 3 1 2 4 5

One routing path assigned for each traffic flow

Example: another possible solution

Example network:



All links with 10 Gbps of capacity

Traffic flows:

t	o_t	d_t	b_t (Gbps)	$\underline{b}_t$ (Gbps)
1	1	4	1.0	1.0
2	1	5	2.0	2.0
3	2	4	3.0	3.0
4	3	5	2.0	2.0

Flow 1:

Path 1 = 1 2 4
Path 2 = 1 3 2 4
Path 3 = 1 5 4
Path 4 = 1 2 5 4
Path 5 = 1 3 2 5 4
Path 6 = 1 5 2 4

Flow 2:

Path 1 = 1 5
Path 2 = 1 2 5
Path 3 = 1 2 4 5
Path 4 = 1 3 2 5
Path 5 = 1 3 2 4 5

Flow 3:

Path 1 = 2 4
Path 2 = 2 5 4
Path 3 = 2 1 5 4
Path 4 = 2 3 1 5 4

Flow 4:

Path 1 = 3 2 5
Path 2 = 3 2 4 5
Path 3 = 3 1 5
Path 4 = 3 1 2 5
Path 5 = 3 2 1 5
Path 6 = 3 1 2 4 5

Different assignments provide different solutions

Traffic engineering objectives

The traffic engineering task aims to:

- optimize at least one parameter related with either the performance or the operational cost of the network;
- optionally, guarantee (maximum or minimum) values for other parameters.

Examples of optimization parameters:

- the average service packet delay (to minimize the delay performance of the service);
- the worst average packet delay among all traffic flows (to minimize the delay performance fairness among all traffic flows);
- the worst link load (to maximize the robustness of the network to unpredictable traffic growth);
- the energy consumption of the network (to minimize operational costs).

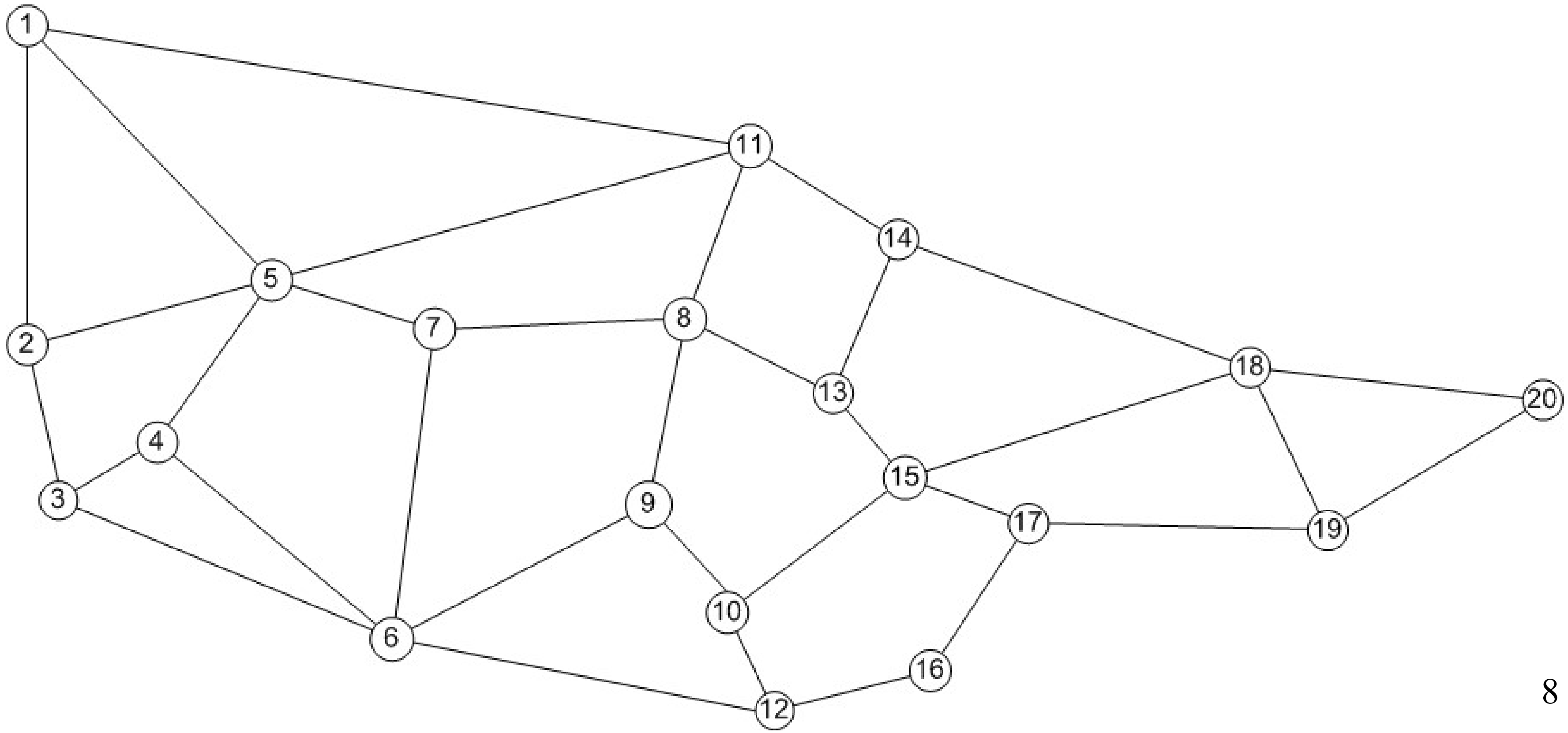
The best traffic engineering solution depends on the optimization objective of interest. Different optimization objectives might be conflicting:

  - for example, to reduce the energy consumption, more links must be put in sleeping mode; consequently, the same traffic is concentrated in less links and the worst link load increases.

Example - network

Consider the following network with 20 routers and 33 links where all links have a capacity of 1 Gbps.

The length of the links varies between 88 km (between nodes 10 and 12) and 759 km (between nodes 1 and 11) and the link propagation delay is determined by the speed of light over a optical fibre (approximately 2 x 10^8 meters/s).



Example – traffic flow matrix

Consider the following flow matrix (values in Mbps) where the packets of all flows are exponentially distributed with an average packet size *B* = 1000 Bytes:

0.0	47.7	64.4	13.6	10.6	45.5	10.6	12.9	9.8	9.8	13.6	11.4	11.4	41.7	12.9	10.6	9.1	12.9	11.4	22.0
47.0	0.0	68.2	32.6	33.3	189.4	59.8	47.0	49.2	38.6	59.1	53.0	38.6	212.1	31.8	48.5	40.9	43.9	54.5	74.2
65.9	69.7	0.0	8.3	13.6	53.0	13.6	14.4	18.9	14.4	11.4	36.4	12.9	63.6	13.6	14.4	11.4	13.6	15.9	22.7
13.6	46.2	13.6	0.0	12.9	31.1	14.4	12.1	11.4	12.9	31.1	12.1	9.1	31.8	11.4	10.6	14.4	12.9	17.4	24.2
7.6	31.8	10.6	14.4	0.0	55.3	11.4	9.1	9.8	12.9	36.4	11.4	12.9	46.2	12.9	7.6	12.1	15.9	16.7	28.0
52.3	174.2	53.8	55.3	38.6	0.0	39.4	47.0	41.7	40.9	53.8	44.7	42.4	212.1	40.9	71.2	62.9	46.2	56.8	72.0
13.6	32.6	11.4	11.4	12.9	54.5	0.0	7.6	12.1	12.9	12.1	14.4	56.8	55.3	9.8	9.1	13.6	15.9	9.8	21.2
13.6	47.0	14.4	11.4	9.8	37.9	10.6	0.0	12.1	9.1	11.4	13.6	12.1	57.6	9.8	10.6	9.8	17.4	12.9	34.1
9.8	33.3	16.7	12.1	14.4	38.6	12.9	9.8	0.0	11.4	13.6	9.1	13.6	56.1	11.4	13.6	12.1	15.2	18.9	18.9
12.9	53.0	10.6	9.8	11.4	46.2	13.6	10.6	10.6	0.0	9.1	9.8	11.4	42.4	10.6	11.4	10.6	15.9	10.6	25.8
9.1	36.4	9.8	31.1	33.3	40.9	9.8	10.6	12.1	14.4	0.0	9.8	10.6	47.7	9.1	11.4	8.3	9.8	12.9	32.6
10.6	61.4	35.6	9.8	9.8	59.8	14.4	8.3	8.3	10.6	12.1	0.0	9.8	33.3	9.8	28.0	9.8	9.1	14.4	25.0
10.6	32.6	9.1	12.1	9.1	35.6	59.8	10.6	9.8	14.4	9.8	13.6	0.0	41.7	11.4	12.9	13.6	15.9	16.7	40.9
40.9	181.8	49.2	56.1	42.4	189.4	55.3	64.4	57.6	31.8	31.8	33.3	46.2	0.0	40.9	57.6	40.2	48.5	51.5	69.7
12.1	37.1	10.6	9.8	10.6	37.1	8.3	14.4	8.3	10.6	9.8	12.1	11.4	47.7	0.0	11.4	10.6	10.6	9.1	28.8
10.6	47.7	9.8	11.4	11.4	44.7	9.8	11.4	10.6	9.1	9.1	12.9	9.1	56.8	14.4	0.0	11.4	9.1	12.9	30.3
13.6	34.1	10.6	10.6	13.6	55.3	12.1	12.9	9.8	11.4	10.6	12.9	9.1	44.7	10.6	9.1	0.0	10.6	9.8	20.5
13.6	40.9	11.4	9.8	18.9	40.9	11.4	18.2	13.6	18.9	12.9	10.6	17.4	40.9	11.4	12.9	9.8	0.0	34.1	24.2
7.6	49.2	18.9	15.9	12.9	53.0	12.1	9.8	15.9	13.6	15.2	10.6	18.9	47.0	12.9	9.8	9.8	30.3	0.0	18.9
23.5	68.2	26.5	28.8	34.1	65.9	25.8	30.3	15.9	22.7	30.3	28.0	34.1	65.2	34.1	25.8	24.2	21.2	15.9	0.0

Example – one possible solution

One possible solution is to route each traffic flow $t \in T$ by the routing path with the shortest length (minimizing, in this way, the propagation delay of each flow).

Using the Kleinrock approximation, we obtain the following performance parameters:

Worst average packet delay = 6.06 ms

Worst link load = 99.3%

Number of active links = 33 out of 33

However, it is possible to obtain better traffic engineering solutions through appropriate optimization algorithms.

Example – optimal solutions

Minimization of the worst average packet delay:

Worst average packet delay = 5.21 ms

Worst link load = 93.6%

Number of active links = 33 out of 33

Minimization of the worst link load:

Worst average packet delay = 8.63 ms

Worst link load = 69.9%

Number of active links = 33 out of 33

Minimization of the number of active links:

Worst average packet delay = 10.54 ms

Worst link load = 82.4%

Number of active links = 26 out of 33

<u>Conclusion</u>:

- Each traffic engineering solution is a different trade-off between the different optimization objectives.
- It is up to the operator to select the best routing solution.

Optimization methods

Exact methods

- Based on mathematical models (for example, Integer Linear Programming)
- In the general case, computationally hard
- Theoretically, they are able to compute the optimal solutions
- Inefficient for large problem instances (they either take too long to even compute feasible solutions or finish due to out-of-memory)

Heuristic methods

- Based on simple programming algorithms
- Easy to implement and quick to find solutions
- Do not guarantee optimality
- For larger runtimes, they find better solutions (than exact methods)
- Efficient for large problem instances

Heuristic method versus heuristic algorithm

Heuristic method: a generic approach to search for good solutions that can be applied to any optimization problem.

Heuristic algorithm: an optimization algorithm that has resulted from applying an heuristic method to a particular optimization problem.

Many heuristic methods (usually, also the simplest ones) are based on two algorithmic strategies:

1. To build a solution starting from the scratch.
   - Examples: *random*, *greedy, greedy randomized, etc…*

2. To get a better solution from a known solution.
   - Examples: *hill climbing, tabu search, simulated annealing, etc...* (we will address only the hill climbing strategy).

Building a solution from the scratch

Building one solution from the scratch (I)

1. Random strategy:

- The solution is built by assigning a random routing path $p \in P_t$ for each flow $t \in T$
- We might obtain better solutions if we consider higher probabilities to routing paths $p \in P_t$ with "better characteristics"
    - For example, paths with a smaller number of links, paths containing links of larger capacity, etc…

2. Greedy strategy:

- Start by considering the network without any routing path
- Then, for each flow $t \in T$:
    - assign the first routing path $p \in P_t$ that, together with the previous assigned routing paths, gives the best objective function value

Building one solution from the scratch (II)

3. Greedy randomized strategy:

The aim is to obtain a different solution on different runs.

First alternative:

- First, choose a random order of the flows $t \in T$
- Then, apply the greedy strategy (previous slide) by the chosen order

Second alternative:

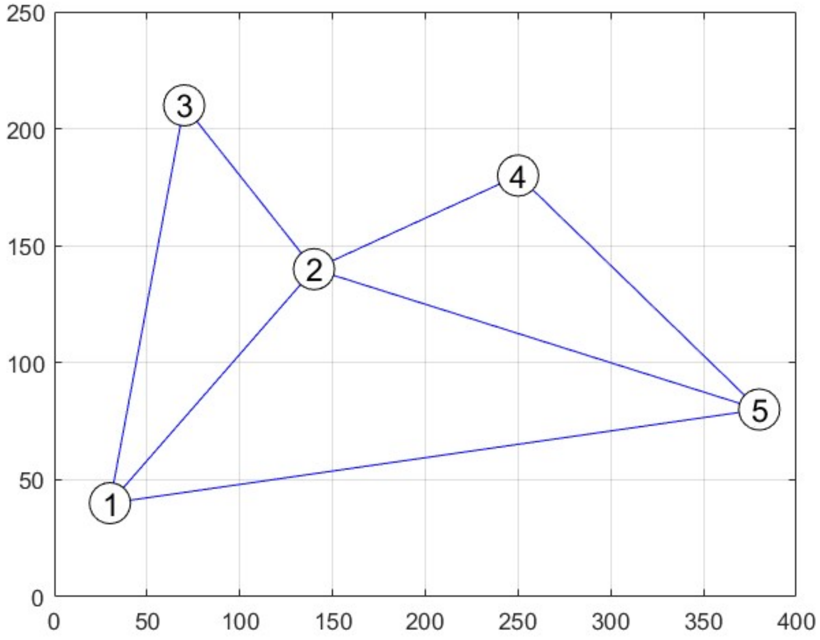
- Start by considering the network without any routing path
- Then, for each flow $t \in T$:
  - compute the α routing paths of P_t that, together with the previous assigned routing paths, give the best objective function values
    - α is a parameter of the algorithm
  - assign randomly one of the previous α routing paths to flow $t \in T$

Third alternative:

- To combine the 2 previous alternatives

Minimizing the worst link load: <u>greedy strategy</u>

Example network:



All links with 10 Gbps of capacity

Traffic flows:

t	o_t	d_t	b_t (Gbps)	$\underline{b}_t$ (Gbps)
1	1	4	1.0	1.0
2	1	5	2.0	2.0
3	2	4	3.0	3.0
4	3	5	2.0	2.0

Flow 1:

Path 1 = 1 2 4
Path 2 = 1 3 2 4
Path 3 = 1 5 4
Path 4 = 1 2 5 4
Path 5 = 1 3 2 5 4
Path 6 = 1 5 2 4

Flow 2:

Path 1 = 1 5
Path 2 = 1 2 5
Path 3 = 1 2 4 5
Path 4 = 1 3 2 5
Path 5 = 1 3 2 4 5

Flow 3:

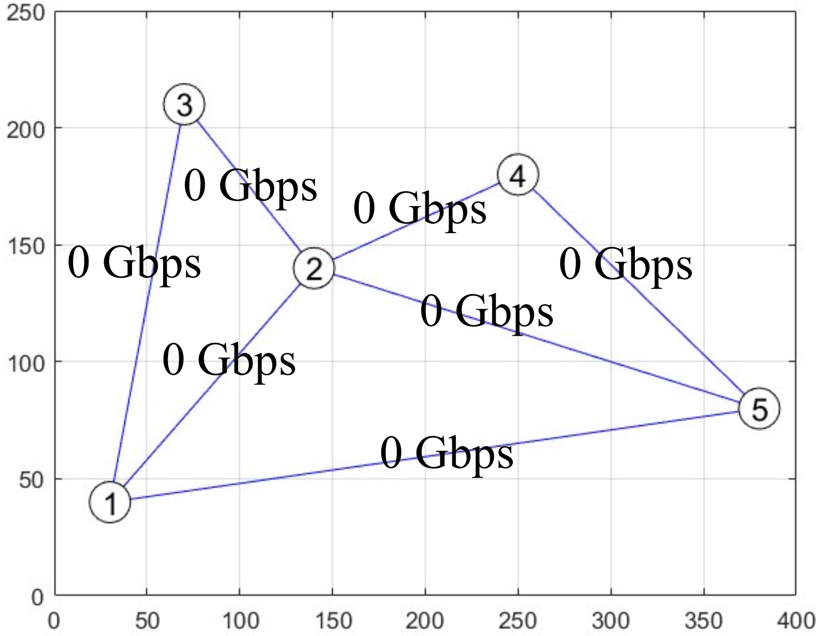
Path 1 = 2 4
Path 2 = 2 5 4
Path 3 = 2 1 5 4
Path 4 = 2 3 1 5 4

Flow 4:

Path 1 = 3 2 5
Path 2 = 3 2 4 5
Path 3 = 3 1 5
Path 4 = 3 1 2 5
Path 5 = 3 2 1 5
Path 6 = 3 1 2 4 5

Greedy strategy: step 1

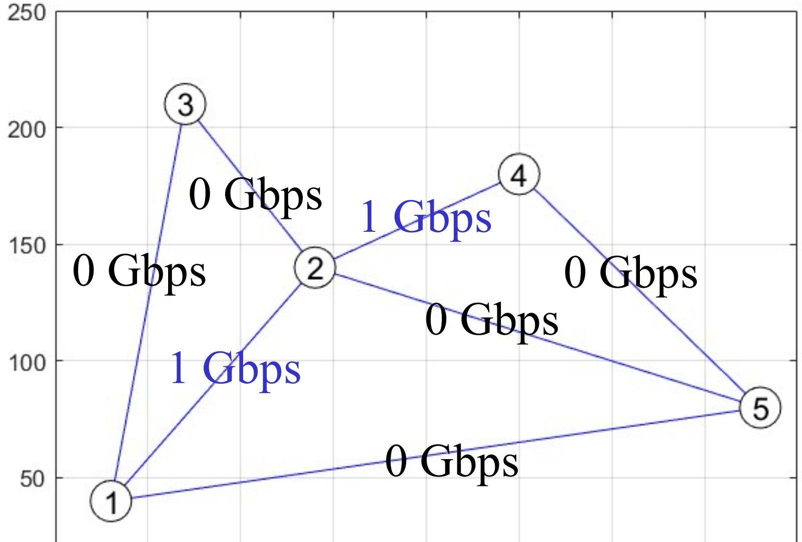
Current solution:



Traffic flows:

t	o_t	d_t	$\overline{b}_t$ (Gbps)	$\underline{b}_t$ (Gbps)
1	1	4	1.0	1.0
2	1	5	2.0	2.0
3	2	4	3.0	3.0
4	3	5	2.0	2.0

Updated solution:



Flow 1:

Path 1 = 1 2 4
Path 2 = 1 3 2 4
Path 3 = 1 5 4
Path 4 = 1 2 5 4
Path 5 = 1 3 2 5 4
Path 6 = 1 5 2 4

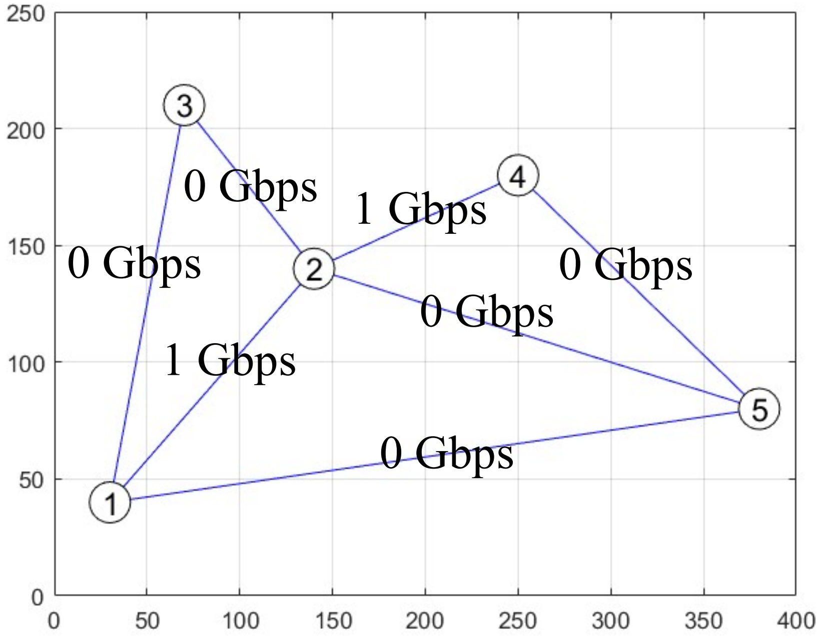
Selected path:

Path 1 = 1 2 4

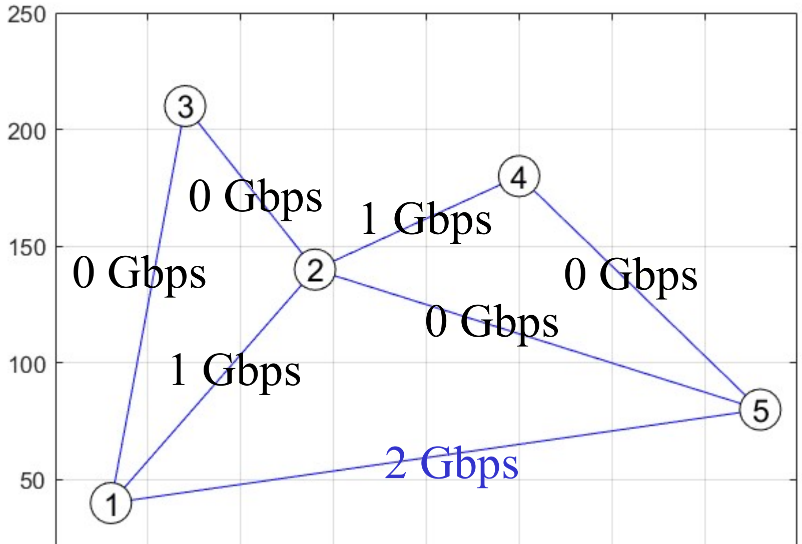
Path minimizing the worst link load in the updated solution

Greedy strategy: step 2

Current solution:



Updated solution:



Traffic flows:

t	o_t	d_t	b_t (Gbps)	$\underline{b}_t$ (Gbps)
1	1	4	1.0	1.0
2	1	5	2.0	2.0
3	2	4	3.0	3.0
4	3	5	2.0	2.0

Flow 2:

Path 1 = 1 5
Path 2 = 1 2 5
Path 3 = 1 2 4 5
Path 4 = 1 3 2 5
Path 5 = 1 3 2 4 5

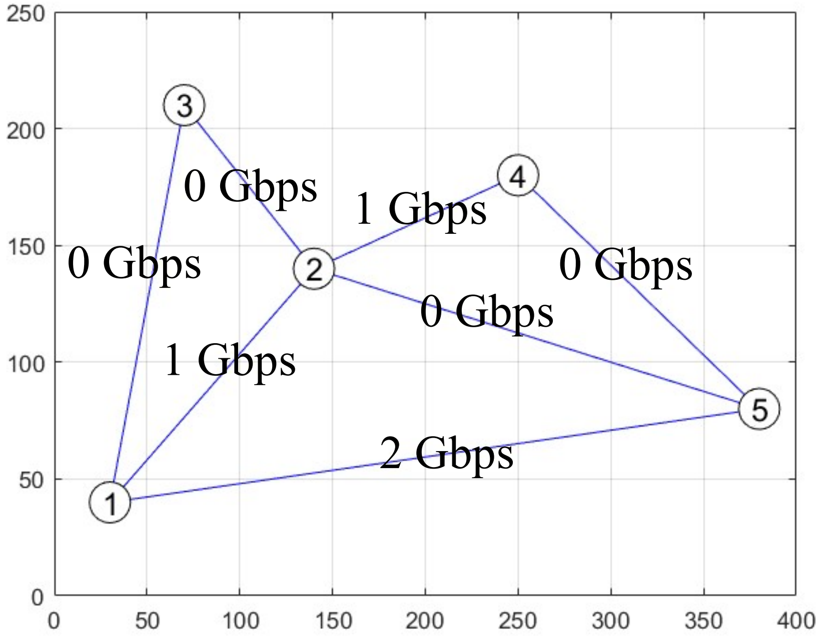
Selected path:

Path 1 = 1 5

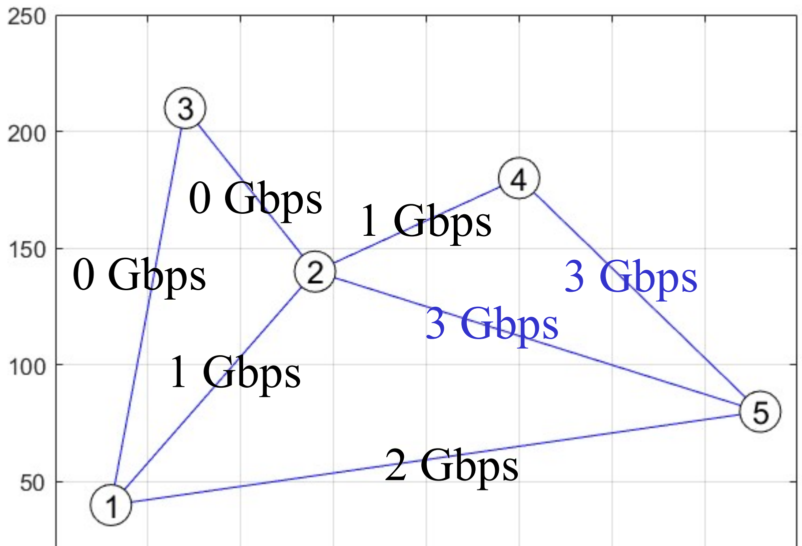
Path minimizing the worst link load in the updated solution

Greedy strategy: step 3

Current solution:



Updated solution:



Traffic flows:

t	o_t	d_t	b_t (Gbps)	$\underline{b}_t$ (Gbps)
1	1	4	1.0	1.0
2	1	5	2.0	2.0
3	2	4	3.0	3.0
4	3	5	2.0	2.0

Flow 3:

Path 1 = 2 4
Path 2 = 2 5 4
Path 3 = 2 1 5 4
Path 4 = 2 3 1 5 4

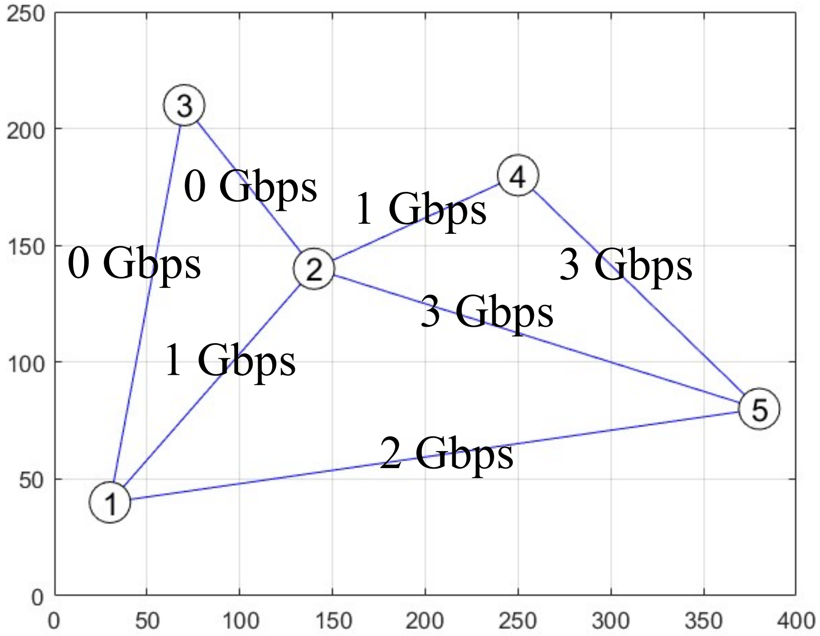
Selected path:

Path 2 = 2 5 4

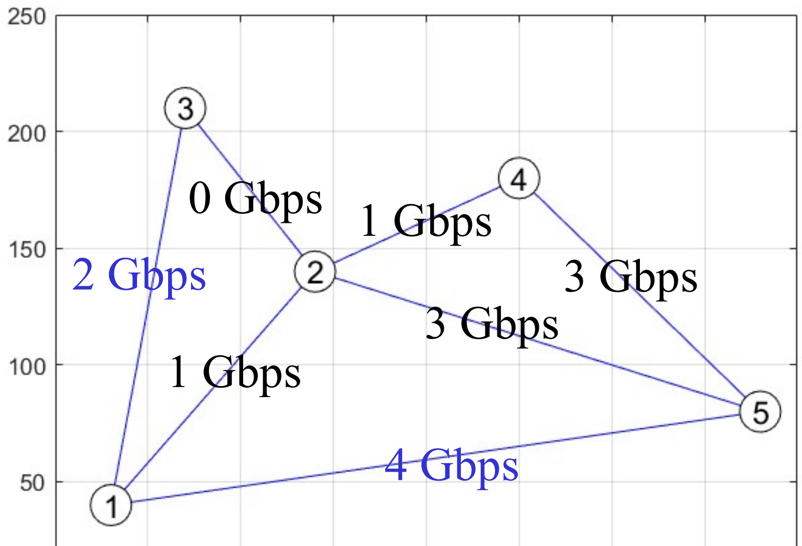
Path minimizing the worst link load in the updated solution

Greedy strategy: step 4

Current solution:



Updated solution:



Traffic flows:

t	o_t	d_t	b_t (Gbps)	$\underline{b}_t$ (Gbps)
1	1	4	1.0	1.0
2	1	5	2.0	2.0
3	2	4	3.0	3.0
4	3	5	2.0	2.0

Flow 4:

Path 1 = 3 2 5
Path 2 = 3 2 4 5
Path 3 = 3 1 5
Path 4 = 3 1 2 5
Path 5 = 3 2 1 5
Path 6 = 3 1 2 4 5

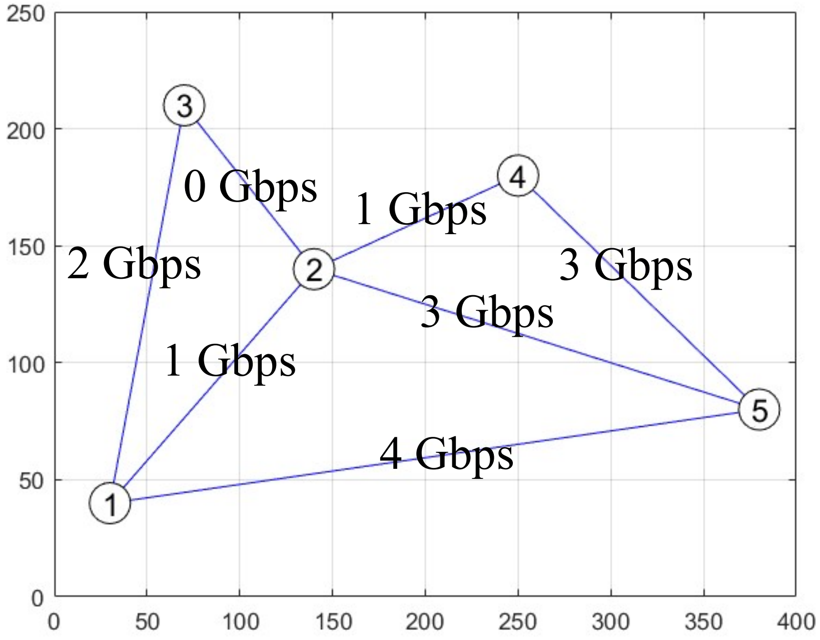
Selected path:

Path 3 = 3 1 5

Path minimizing the worst link load in the updated solution

Greedy strategy: FINAL SOLUTION

Current solution:



All links with 10 Gbps of capacity

Traffic flows:

t	o_t	d_t	b_t (Gbps)	$\underline{b}_t$ (Gbps)
1	1	4	1.0	1.0
2	1	5	2.0	2.0
3	2	4	3.0	3.0
4	3	5	2.0	2.0

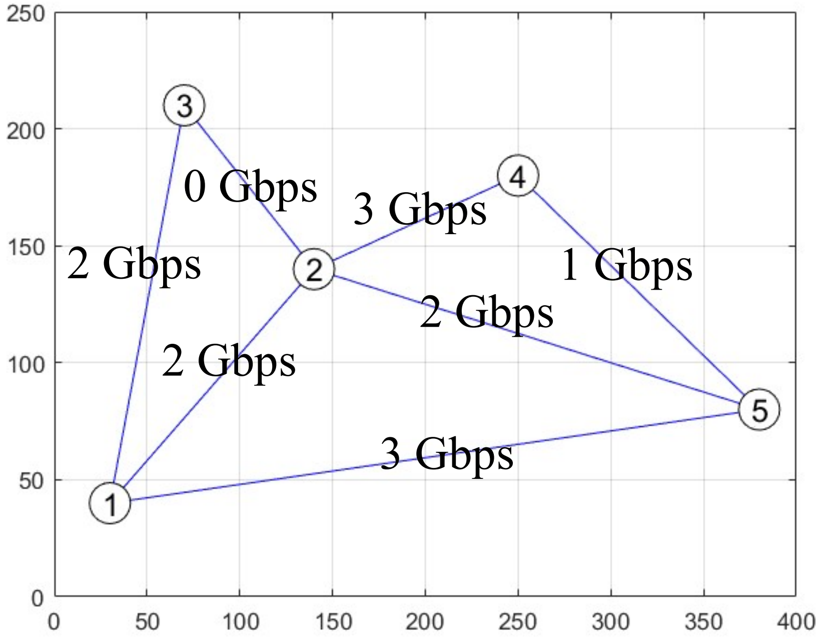
FINAL SOLUTION :

Flow 1:
Path 1 = 1 2 4
Flow 2:
Path 1 = 1 5
Flow 3:
Path 3 = 2 5 4
Flow 4:
Path 3 = 3 1 5

Worst link load → 4 Gbps (= 40%)
Link(s) with the worst link load → (1,5) and (5,1)
Unused links → {2,3}

Greedy randomized strategy

Current solution:



All links with 10 Gbps of capacity

Traffic flows:

t	o_t	d_t	b_t (Gbps)	$\underline{b}_t$ (Gbps)
1	1	4	1.0	1.0
2	1	5	2.0	2.0
3	2	4	3.0	3.0
4	3	5	2.0	2.0

Taking the random order 3 → 1 → 2 → 4 :

Flow 3:
Path 1 = 2 4
Flow 1:
Path 3 = 1 5 4
Flow 2:
Path 3 = 1 2 5
Flow 4:
Path 3 = 3 1 5

Worst link load → 3 Gbps (= 30%)
Link(s) with the worst link load → (1,5), (5,1), (2,4) and (4,2)
Unused links → {2,3}

Optimization algorithm

- In a problem aiming to minimize function $F(x)$, it works as follows:

$$f_{best} = +\infty$$

repeat

$x = BuildSolution\ ()$

Random strategy or
Greedy Randomized strategy

$f = F(x)$

if $f < f_{best}$ **then**

$x_{best} = x$

$f_{best} = f$

endif

until Stopping Criteria is met

If the aim is to maximize $F(x)$

$f_{best} = -\infty$

$f > f_{best}$

- Examples of Stopping Criteria:
  - Run a predefined time duration
  - Run a predefined number of iterations
  - Run until f_{best} not improving a predefined number of iterations