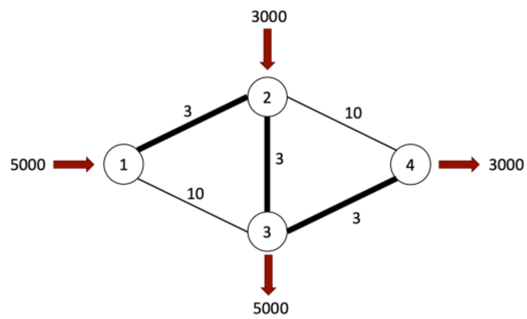
Shanghai Jiao Tong University ECE4530J

Mini Project 2: Due 2022.7.7

Consider the undirected network in the figure below. Links (1,2), (2,3), (3,4) are highways with a capacity of 6000 veh/hr and a travel time of 3 min. Links (1,3), (2,4) are local streets with a capacity of 3000 veh/hr and a travel time of 10 min. The traffic demand is indicated in the figure as well (unit: veh/hr). We want to allocate the traffic flows to minimize the average travel time for all vehicles.



a) Suppose that we do not differentiate traffic according to their origin-destination (OD) information. Formulate the min-cost flow problem. Solve it using a coding language of your choice.

b) Now, suppose that every vehicle entering the network at node 1 (resp. 2) must exit through node 3 (resp. 4). Formulate the min-cost flow problem. Solve it using a coding language of your choice.

Tools and tutorials for linear optimization:

Python:

Scipy is a very powerful Python library to solve optimization problem. Below is a vivid example about linear optimization. For other details, please check the official guidance here:
https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.linprog.html

Example:

Suppose we are to solve the following linear optimization problem

$$\begin{aligned} \min_{x_0, x_1} \quad & -x_0 + 4x_1 \\ \text{such that} \quad & -3x_0 + x_1 \le 6, \\ & -x_0 - 2x_1 \ge -4, \\ & x_1 \ge -3. \end{aligned}$$

Firstly, convert the above problem into a “standard” formulation as follows:

$$\begin{aligned} \min_{x_0, x_1} \quad & -x_0 + 4x_1 \\ \text{s.t.} \quad & -3x_0 + x_1 \le 6 \\ & x_0 + 2x_1 \le 4 \\ & x_1 \ge -3 \end{aligned}$$

"Standard" formulation requires: i) the objective is minimizing (if your objective function is to maximize f(x), it is equivalent to minimize -f(x)); ii) the constraints should be "less than".

```
c = [-1, 4] # coefficient parameters of objective function

A = [[-3, 1], [1, 2]]  # Constraint matrix in the "standard" formulation

b = [6, 4]  # right side coefficients in the constraints

x0_bounds = (None, None)  # we do not have constraint about x_0's range
x1_bounds = (-3, None)   # x_1 must be greater than or equal to -3

from scipy.optimize import linprog  # import the scipy libray

res = linprog(c, A_ub=A, b_ub=b, bounds=[x0_bounds, x1_bounds])
print(res)  # The most important outputs are second line and last line
     con: array([], dtype=float64)
     fun: -21.99999984082494 # optimal objective function
 message: 'Optimization terminated successfully.'
     nit: 6 # may vary
   slack: array([3.89999997e+01, 8.46872439e-08] # may vary
  status: 0
 success: True
       x: array([ 9.99999989, -2.99999999]) # optimal x_0 and x_1
```

MATLAB:

Tutorials of using Matlab to solve linear optimization problem:

Official guidance: https://www.mathworks.com/help/optim/ug/linprog.html
A great video tutorial: https://www.youtube.com/watch?v=kavYLZatz44

For students whose future research direction is around optimization, highly recommend learning to use "Gurobi", which is the most powerful optimization solver. "Gurobi" is compatible with Python, Matlab, C++, Java, etc.

You can get a free Gurobi license via your SJTU email. For details, check the Gurobi website: https://www.gurobi.com/