**CE 272**
**Traffic Network Equilibrium**

Programming Task 3
Informal Evaluation Due May 21

**General Instructions:** This task is the third of four tasks in the computer programming project and involves implementing the MSA algorithm. Share the link to your Colab with the TA (Helen Thomas).

**Demand Data for the Project:** The repository has a file `networkname_trips.tntp` containing the demand between each OD pair. The first few lines indicate the number of zones and total demand (which could be used to verify if the data was properly read by the code).

```
<NUMBER OF ZONES> 24
<TOTAL OD FLOW> 360600.0
<END OF METADATA>
```

In remaining part of the file, for each origin, the demand to all other destinations are listed.

```
Origin        1
   1 :      0.0;     2 :    100.0;     3 :    100.0;     4 :    500.0;     5 :    200.0;
   6 :    300.0;     7 :    500.0;     8 :    800.0;     9 :    500.0;    10 :   1300.0;
  11 :    500.0;    12 :    200.0;    13 :    500.0;    14 :    300.0;    15 :    500.0;
  16 :    500.0;    17 :    400.0;    18 :    100.0;    19 :    300.0;    20 :    300.0;
  21 :    100.0;    22 :    400.0;    23 :    300.0;    24 :    100.0;
```

**Task Objectives:**

1. Select the `SiouxFalls` network. Initialize a (Number of Zones) × (Number of Zones) matrix called `demand` and store the associated data from `SiouxFalls_trips.tntp`.
2. To the `arcs` class, add three new members to store equilibrium flow and travel time information: `flow`, `aon flow`, and `time`. Use `flow` and `aon flow` to store the current and target arc flow (from the all-or-nothing assignment). The variable `time` can be used to store the travel time associated with the `flow` value.
3. Using members of the arc class and the following BPR function, write a function that updates the arc times from the arc flows.

$$\text{time} = \text{free flow time}\left(1 + \text{B}\left(\frac{\text{flow}}{\text{capacity}}\right)^{power}\right)$$

4. Write a function to estimate the all-or-nothing flows by calling one of the shortest path algorithms (`label correcting` or `label setting` functions) from the previous task.
5. Write another function `calculate gap` that computes AEC and relative gap.
6. Finally, code the `msa` algorithm using the above functions and terminate when the relative gap is less than $10^{-4}$.
7. Test your code on `Eastern-Massachusetts`, `Chicago-Sketch`, and `Anaheim`.