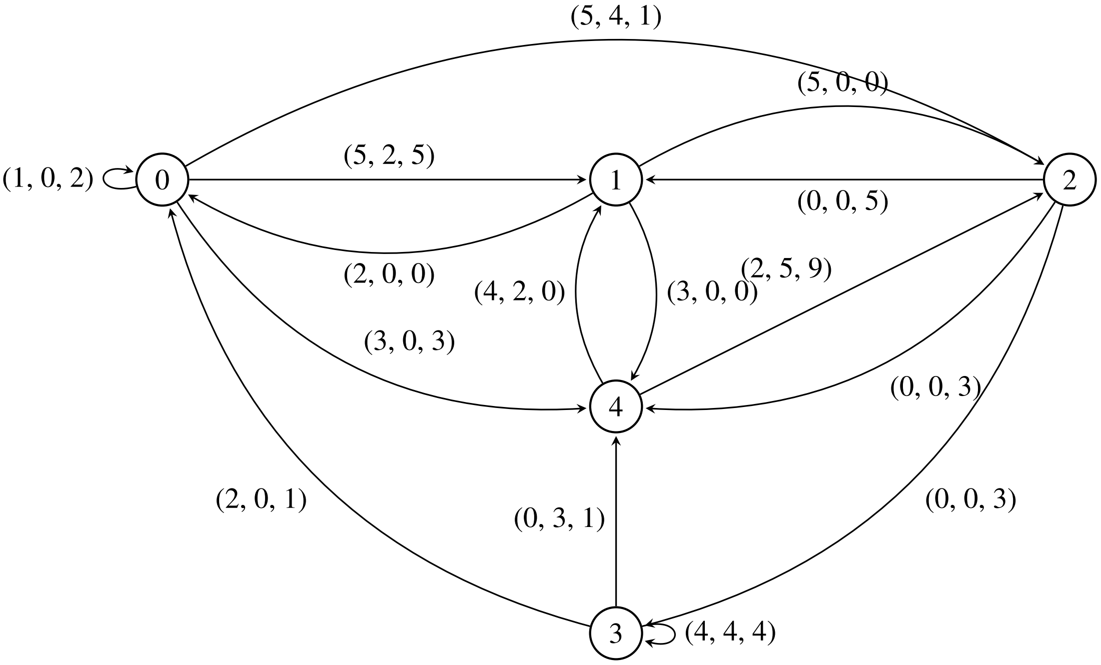
Workshop 01: Numpy

Mr. Abdelkrime ARIES

Objective: Learn numpy (which will be used afterwards to implement basic ML algorithms)
Demo: `https://github.com/projeduc/ESI_ML/blob/main/demos/Init/ML_numpy.ipynb`
Numpy doc: `https://numpy.org/doc/stable/`

We want to study users' popularity on Twitter. To do this, we will analyze the interactions (Reply, Retweet, Like) between 5 users. The number of tweets by each user is represented by the vector: **[2, 5, 10, 4, 3]**. Responses are not considered as tweets. The following graph represents interactions in the form of (Reply, Retweet, Like) between users. When there is no edge from one node to another, this means that all three interactions are (0, 0, 0).



- Represent the interactions as three 5x5 matrices.

Engagement Statistics

Here, we want to extract some statistics on user engagement with others. In this case, we are interested in outgoing edges.

1. Find the user who replies more than others. *It should be user 0*
2. Find the user who retweets more than others. *It should be user 4*
3. Find users who replied or retweeted at least once. *It should be all users except user 2*
4. Display the number of likes each user has given to their own tweets. *It should be [2., 0., 0., 4., 0.]*
5. Display the number of likes each user has given to tweets other than their own. *It should be [9., 0., 11., 2., 9.]*
6. Find users who gave likes to their own tweets more than they gave to others'. *It should be just user 3*
7. Calculate their number. *Just 1*
8. Display the number of interactions each user has made. *It should be [31, 10, 11, 15, 22]*

Interaction Statistics

1. Display the maximum number of likes each user has received (from users other than themselves). *To eliminate the diagonal, we can create a 5x5 diagonal matrix and invert the values. Then, apply element-wise multiplication. It should be [2, 5, 9, 4, 3]*
2. Find users who have received likes for all their tweets by at least one person other than themselves. For example, User 1 has posted 5 tweets and there is at least one other user who has liked all 5 of these tweets (in this case, Users 0 and 2). *It should be all users except user 2*
3. Calculate the number of incoming edges (having at least one interaction). *It should be [3, 3, 3, 2, 4]*
4. Calculate the number of outgoing edges (having at least one interaction). *It should be [4, 3, 3, 3, 2]*
5. Find nodes where the number of incoming edges is equal to the number of outgoing edges. *It should be users 1 and 2*
6. Display the likes they have given to each other. *It should be [[0, 0], [5, 0]] or something equivalent*

Popularity

- We are not interested in reflexive interactions (e.g., someone liking their own tweets), which are represented by the diagonal for Likes and Replies. However, we are interested in reflexive Retweets. *To eliminate the diagonal, we can create a 5x5 diagonal matrix and invert the values. Then, apply element-wise multiplication.*
- We want to weight the matrices as follows:
  - Reply Score: We assume that Replies are generally objections; therefore, they have a negative effect. For each interaction, if the number of Replies received from a person is more than their Likes, we multiply the number of Replies by (-0.5). If it is equal, we multiply their own Likes by (-0.25). Otherwise, it has no effect (0). **HINT: Use numpy.where**
  - Retweet Score: We assume that Retweets are generally neutral; therefore, we do not apply any transformation on these interactions. However, if someone retweets their own tweets, they will be penalized by (-0.25) for each retweet.
  - Like Score: We assume that Likes have a positive effect on popularity. For each interaction, if the number of Likes received from a person is greater than half of the tweets published, we multiply it by (1.5). Otherwise, the score is the number of Likes.
- Calculate the popularity scores of each user for his interactions with each user. *It should be*

```
[[ 0.        ,  2.5       , -0.5       ,  0.        ,  1.5       ],
 [-0.33333333,  0.        , -0.83333333,  0.        , -0.5       ],
 [ 0.        ,  2.5       ,  0.        ,  1.5       ,  1.5       ],
 [ 0.        ,  0.        ,  0.        ,  0.        ,  0.33333333],
 [ 0.        , -0.66666667,  5.16666667,  0.        ,  0.        ]]
```

- Calculate the final score which is the average of the scores from different interactions. *It should be [-0.06666667, 0.86666667, 0.76666667, 0.3 , 0.56666667]*