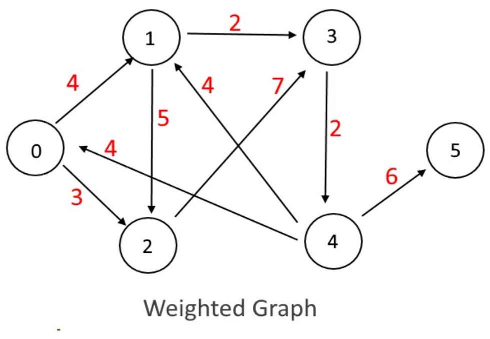
Spark: Single-Source Shortest Path

Problem Definition:

Consider a graph consisting of vertices/nodes, edges and weights for the edges.



Weighted Graph

(https://i1.wp.com/algorithms.tutorialhorizon.com/files/2018/03/Weighted-Graph.png?ssl=1)

A path in a graph can be defined as the set of consecutive nodes such that there is an edge from one node to the next node in the sequence. The shortest path between two nodes can be defined as the path that has the minimum total weight of the edges along the path. For example, the shortest path from **Node 1** to **Node 4** is Node1-> Node3 -> Node4 with the distance of 4.

In this assignment, you are asked to calculate the distance from a starting node (e.g., N0) to all other nodes.

Please consider the following tips while implementing your solution:

1. In this assignment we assume there is no abandoned node, meaning that each node at least has one edge to another node.
2. The minimum weight of an edge between two individual nodes is 1.
3. You need to use Spark Core (not GraphX or similar solutions) to implement a solution for this given problem.
4. The output must be sorted (ascending) by length of shortest path.
5. The output should be written into a txt file.
6. The output should be formatted as follow (as shown in the example below) containing 3 columns, comma delimited : first column contains the destination node, followed by the length of the shortest path in the second column, and the actual shortest path from the starting node to the destination node.
7. You should NOT assume that the starting node is always N0 (it can be any given node).
8. You can download a sample input (https://github.com/mysilver/COMP9313/blob/master/graph.txt) , and expected output (https://github.com/mysilver/COMP9313/blob/master/shortest-pathes) .
9. if the start-node is the last node of the graph (e.g.,N5), which means it has no way to get to another nodes, the output should be -1 for all the nodes as shown below :

```
N0,-1,
N1,-1,
...
```

Input Format:

You can assume that the input represents a connected directed graph.
The input is formatted as below (represents the same graph as the above image). Each line of the input file represents a vertex of the graph formatted like: starting node, end node, and the distance

```
N0,N1,4
N0,N2,3
N1,N2,2
N1,N3,2
N2,N3,7
N3,N4,2
N4,N0,4
N4,N1,4
N4,N5,6
```

Output Format:

The output should be formats as below. Each line of output represents the distance from the starting node to another node, and it is formatted as: the destination node, shortest distance, and the path from the starting node to the destination node. The file should be sorted by the shortest path:

```
N2,3,N0-N2
N1,4,N0-N1
N3,6,N0-N1-N3
...
```

Submission Guidelines

Submission Deadline:

~~Monday the 18th of November 2019 17:59~~

~~Tuesday the 19th of November 2019 17:59~~

Thursday the 21st of November 2019 17:59

Build your Project

Get and add Spark dependency to your project:

1. Create a new Java project in Eclipse
2. Download Spark-Core.jar https://www.dropbox.com/s/xragkjaio6n9onp/Spark-Core.jar?dl=0 (https://www.dropbox.com/s/xragkjaio6n9onp/Spark-Core.jar?dl=0)
3. Add the dependency to your project: right click on the project; next : Build Path > Add External Archives

Submit your Project

Your code must be included (in its entirety) in the file **AssigTwo{zid}.java** . Any solution that has compilation errors will receive no more than 5 points for the entire assignment.

You need to test your file before submission, to make sure that it can be compiled/run in the terminal of CSE machines:

```
$ javac -cp ".:Spark-Core.jar" AssigTwo{zid}.java
$ java -cp ".:Spark-Core.jar" AssigTwo{zid} STARTING_NODE INPUT_PATH OUTPUT_PATH
```

Log in to any CSE server (e.g. williams or wagner) and use the *give command* below to submit your solution:

```
$ give cs9313 assig2 AssigTwo{zid}.java
```

where you must replace {zid} above with your own zID.

You can also submit your solution using WebCMS, or Give:

https://cgi.cse.unsw.edu.au/~give/Student/give.php (https://cgi.cse.unsw.edu.au/~give/Student/give.php)

If you submit your assignment more than once, we will only consider the last submission. If you face any problem while submitting your code, please e-mail the Course Admin (Maisie Badami, m.badami@student.unsw.edu.au (mailto:m.badami@student.unsw.edu.au))

Assessment

Your source code will be manually inspected and marked based on readability and ease of understanding. We will run your code to verify that it produces correct results. The code documentation (i.e. comments in your source code) is also important. Below, we provide an indicative assessment scheme (maximum mark: 25 points):

1. The output is generated by Apache Spark Core correctly **(12 Marks)**
2. The output is formatted correctly as described in the spec **(3 marks)**
   - the format should be as

```
N2,3,N0-N2
N1,4,N0-N1
```

3. The output must be sorted (ascending) by length of shortest path **(4 Marks)**
4. The code could be executed in CSE machines as described in the specification of the assignment **(2 Marks)**
5. Documentation and code structure. here you can clearly explain you solution in a short paragraph in the beginning of the program (no more than 300 words) and provide comments describing what each class is doing. **(4 Marks)**

~~**10% reduction of your marks for the 1st day, 30% reduction/day for the following days.Late submission penalty**~~

20% reduction of your marks for the 1st day, 40% reduction/day for the following days.Late submission penalty

Plagiarism

This is an *individual assignment* . The work you submit must be your own work. Submission of work partially or completely derived from any other person or jointly written with any other person is not permitted. The penalties for such offence may include negative marks, automatic failure of the course and possibly other academic discipline. Assignment submissions will be examined manually.

Do not provide or show your assignment work to any other person - apart from the teaching staff of this course. If you knowingly provide or show your assignment work to another person for any reason, and work derived from it is submitted, you may be penalized, even if the work was submitted without your knowledge or consent. Pay attention that is also your duty to protect your code artifacts. if you are using any online solution to store your code artifacts (e.g., GitHub) then make sure to keep the reposiroty private and do not share access to anyone.

Reminder: Plagiarism is defined as (https://student.unsw.edu.au/plagiarism) using the words or ideas of others and presenting them as your own. UNSW and CSE treat plagiarism as academic misconduct, which means that it carries penalties as severe as being excluded from further study at UNSW. There are several on-line sources to help you understand what plagiarism is and how it is dealt with at UNSW:

- Plagiarism and Academic Integrity (https://student.unsw.edu.au/plagiarism)
- UNSW Plagiarism Procedure (https://www.gs.unsw.edu.au/policy/documents/plagiarismprocedure.pdf)

Make sure that you read and understand these. Ignorance is not accepted as an excuse for plagiarism. In particular, you are also responsible for ensuring that your assignment files are not accessible by anyone but you by setting the correct permissions in your CSE directory and code repository, if using one (e.g., Github and similar). Note also that plagiarism includes paying or asking another person to do a piece of work for you and then submitting it as your own work.

UNSW has an ongoing commitment to fostering a culture of learning informed by academic integrity. All UNSW staff and students have a responsibility to adhere to this principle of academic integrity. Plagiarism undermines academic integrity and is not tolerated at UNSW.