Western Science

ASSIGNMENT 04

DISTANCE VECTOR ROUTING

Due Date:
December 5, 2024 23:55:59

11d 11h 33m 57s left

ACADEMIC DISHONESTY

If you want to store a PDF version of this assignment, press Ctrl+p *on Windows or* Command+p *on Mac, the print window will appear. Then, select* Save As PDF *from the* ***Destination*** *dropdown. Then click* Save

This file was last modified on 2024-11-15 10:19 am ET

UPDATES and CHANGES

The following are the updates/changes made to this assignment AFTER it was posted:

No changes have been made!

Anouncements and Important Notes

TOP

The following are the announcements and important notes for this assignment AFTER it was posted:

No announcements have been made!

1 Problem Description

Given an `N` x `N` adjacency matrix representing a directed graph or node network each entry represents the weight of the edge connecting a node to another. For example, the entry at row `i` and column `j` represents the weight of the edge connecting node `i` to node `j`.

If the entry is either infinity, it means there is no edge connecting the two nodes. The diagonal entries of the matrix are always zero, as there are no self-loops in the graph. The graph may contain negative cycles.

Your task is to implement the **Bellman-Ford** algorithm for distance vector routing. The algorithm should calculate the shortest distance from each node to every other node in the graph. The distance between two nodes is the sum of the weights of the edges in the shortest path between them.

Each node should have a distance array that stores the shortest distance from that node to every other node in the graph. If a node cannot reach another node, the distance should be set to infinity. If a node gets into a negative cycle during the algorithm, the distance to all other nodes should be set to `None`, for example, if we have a graph with 3 nodes and there is negative cycle while calculating the shortest distance from node 0, the output should be: `Node 0: [None, None, None]`.

Standard Input

- The first line contains an integer `N` (1 ≤ `N` ≤ 50) representing the dimension of the adjacency matrix.
- The following `N*N` lines contain the entries of the adjacency matrix. Each line contains one entry, which can be an integer number (positive or negative) or the string `f` representing infinity.

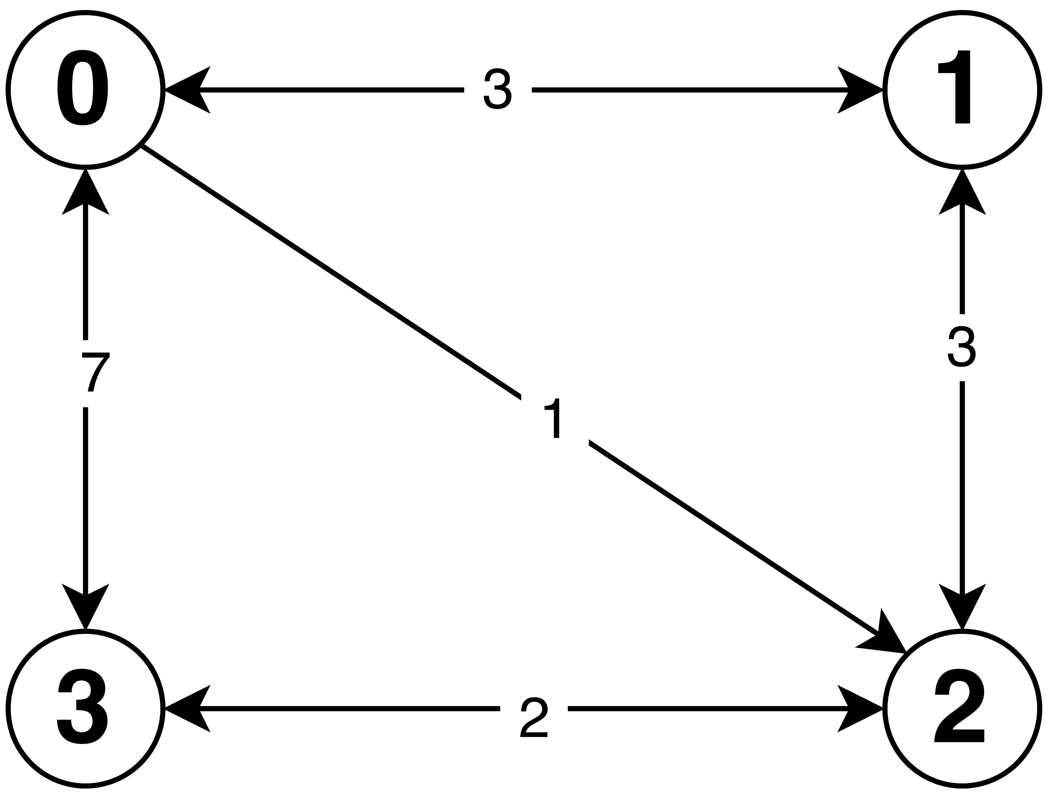
Standard Output

- For each node, print the shortest distance array. Required output format is: `"Node *i*: *distance_array*"` where `i` is the node number and `distance_array` is the shortest distance array from that node to every other node (including itself) in the graph.

- Example

TOP

Consider the following diagram of node network or graph with 4 nodes (0, 1, 2, 3).



The adjacency matrix (intial distance matrix) that represents the graph is as follows:

```
# Node    0       1       2       3
---------------------------------
0    |  0       3       1       7
1    |  3       0       3       ∞
2    |  ∞       3       0       2
3    |  7       ∞       2       0
```

so the input sequence to your program for this graph would be:

4 ↵ 0 ↵ 3 ↵ 1 ↵ 7 ↵ 3 ↵ 0 ↵ 3 ↵ f ↵ f ↵ 3 ↵ 0 ↵ 2 ↵ 7 ↵ f ↵ 2 ↵ 0

The output of your program should be:

```
Node 0: [0, 3, 1, 3]
Node 1: [3, 0, 3, 5]
Node 2: [6, 3, 0, 2]
Node 3: [7, 5, 2, 0]
```

- Submission

REMEMBER!

- You have 4 coupons (for the entire semester) that will be automatically applied when you submit late.
- It is the student's responsibility to ensure the work was submitted and posted in GradeScope.

> - Any assignment not submitted correctly will not be graded.
> - Submissions through the email will not be accepted under any circumstances.
> - Please check this page back whenever an announcement is posted regarding this assignment.
> - Marks will be deducted if you submit anything other than the required Python files.
> - Submit the assignment on time. Late submissions will penalty unless you have enough coupons to be applied (automatically).
> - You may re-submit your code as many times as you like. Gradescope uses your last submission for grading by default. There are no penalties for re-submitting. However, re-submissions that come in after the due date will be considered late.
> - Again, assignments will be run through similarity checking software to check for code similarity. Sharing or copying code in any way is considered plagiarism (Academic dishonesty) and may result in a mark of 0 on the assignment and/or be reported to the Dean's Office. Plagiarism is a serious offense. Work is to be done individually.

1. You must submit one file only to the Assignment submission page on Gradescope. The required file is as follows:

   `distance_vector.py`

2. Make sure that you get the confirmation email after submission.

3. The submission will be using Gradescope Assignment-04 page .