Bellman-Ford Algorithm

Path with Maximum Probability

You are given an undirected weighted graph of n nodes (0-indexed), represented by an edge list where edges[i] = [a, b] is an undirected edge connecting the nodes a and b with a probability of success of traversing that edge succProb[i].

Given two nodes start and end, find the path with the maximum probability of success to go from start to end and return its success probability.

If there is no path from start to end, return 0. Your answer will be accepted if it differs from the correct answer by at most 1e-5.

Example 1:

Input: n=3,edges=[[0,1],[1,2],[0,2]],succProb=[0.5,0.5,0.2],start=0,end=2

Output:0.25000

import heapq

import math

def maxProbability(n,edges,succProb,start,end):

graph=[[]for \_ in range(n)]

for (a, b),prob in zip(edges,succProb):

graph[a].append((b,prob))

graph[b].append((a,prob))

max\_prob=[0]\*n

max\_prob[start]=1

pq=[(-1,start)]

while pq:

current\_prob,node=heapq.heappop(pq)

current\_prob=-current\_prob

if node==end:

return current\_prob

for neighbor,prob in graph[node]:

new\_prob=current\_prob\*prob

if new\_prob>max\_prob[neighbor]:

max\_prob[neighbor]=new\_prob

heapq.heappush(pq,(-new\_prob,neighbor))

return 0

n=3

edges=[[0,1],[1,2],[0,2]]

succProb=[0.5,0.5,0.2]

start=0

end=2

print(f"{maxProbability(n,edges,succProb,start,end):.5f}")

Output :

0.25000