Djikstra

import heapq

def dijkstra(graph, start):

# Distance to all nodes initially infinity

distances = {node: float('inf') for node in graph}

distances[start] = 0

# Priority queue to get the node with the smallest distance

pq = [(0, start)]

while pq:

current\_distance, current\_node = heapq.heappop(pq)

# Skip if we already found a better path

if current\_distance > distances[current\_node]:

continue

for neighbor, weight in graph[current\_node]:

distance = current\_distance + weight

# If new distance is smaller, update it

if distance < distances[neighbor]:

distances[neighbor] = distance

heapq.heappush(pq, (distance, neighbor))

return distances

# Define the graph as an adjacency list

graph = {

'A': [('B', 4), ('C', 5)],

'B': [('A', 4), ('C', 11), ('D', 9), ('E', 7)],

'C': [('A', 5), ('B', 11), ('E', 3)],

'D': [('B', 9), ('F', 2)],

'E': [('B', 7), ('C', 3), ('F', 6)],

'F': [('D', 2), ('E', 6)]

}

# Run Dijkstra from source node 'A'

distances = dijkstra(graph, 'A')

print("Shortest distances from A:")

for node in distances:

print(f"{node}: {distances[node]}")

theory

Dijkstra's Algorithm is a Greedy algorithm used to find the shortest path from a source node to all other nodes in a weighted graph. It works only with graphs that have non-negative weights and guarantees the shortest path in terms of total edge weight.

Steps of Dijkstra’s Algorithm:

Initialize:

Set the distance to the source node (start) as 0 and all other nodes as infinity (∞).

Place the source node in a priority queue with a distance of 0.

Process the Priority Queue:

While the priority queue is not empty:

Extract the node with the smallest known distance from the queue.

For the current node, check all its neighbors. Calculate the tentative distance to each neighbor through the current node.

If the calculated distance is smaller than the known distance to the neighbor, update the neighbor’s distance and add it to the priority queue.

Repeat until all nodes are processed or until the destination node (if any) is reached.

Return the shortest distances from the source to all other nodes.