Graphs 4

Friday, January 13th 2023 1:44 PM

Q Given an unweighted graph
Find min path distance from a source node to all other nodes.

→

TC = O(V+E)
SC = O(V)

Simple BFS.

Q Same Question, but weighted
{1, 2, 3}

a)

Remember that edges are in the range {1, 2, 3}

Now do simple BFS on this graph.

Original : V, E

Apply weighting : V' = V + 2E
(worst case)

E' = 3E
→ 2 edges for every edge

TC = O(V'+E') = O(V+2E+3E)
= O(V+E)

Q Same question, weights : 1 - 10^3

Dijkstra's Algorithm

[for edge weights, Dijkstra's]

- Simple source shortest path algorithm

<E,3>
<B,5>
<D,6>
min heap

<A,6>
<B,4>

Requirements of DS
→ Insert
→ Get min
→ Del min
} min heap

Also Distance array

d = [∞] * N

heap <Int, Int> → priority queue
node dist
(pair)

heap.push ((0,S))

while heap:
    p = heap.pop()
    dist, v = p
    if d[v] == ∞:
        d[v] = dist
    for w in adj[v]:
        if d[w] == ∞:
            heap.push ((dist + w[v→w], w))
                edge wt

return d

TC = O(V + E log E)

total operations on the heap = 2E

since many vertices can be added to the heap at once
So, roughly
O(nlog n) = O (2E log 2E)
= O(E log E)

One step away can into have some d = d[v], +1 ;
- one group is detected
- Group is detected

Floyd-Warshall Algorithm

- All pair shortest path algorithm
- -ve weights are allowed
- works on directed graphs
(If undirected → convert edge to directed

- It works is applied on all pairs.
It would be V(V+E)

Relaxation of an Edge

d (0,3) = min (d(0,3), d(0,1) + d(1,3))
= min (5, 2+3)
= 5

d(0,3) = min (d(0,3), d(0,1) + d(1,3))
= min (∞ , 4 + 3)
= 7

Here, our edge 0→3 is weaker as it doesn't have one with dist.
So it is relaxed

d[0,3] = 3
= 3 + (-3) = 0 + (-3)
= 0.

But with cycle, we go and dist would keep decreasing.
We can say moreover we shortest d[0,3]

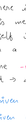
Problem In our cycle Overall output is negative, called as negative wt cycle

Floyd-Warshall can detect negative cycles

Idea For all nodes, consider them as intermediate nodes.
Find ways to reduce as many edges as possible

v : intermediate node

d(u,v) = min (d(u,v) , d(u,v) + d(v,v))
without Int | with Int



	1	2	3	4
1	0	3	-4	∞
2	6	0	∞	2
3	∞	5	0	∞
4	∞	∞	2	0

Now consider every node as Int & see if we can relax every other node

for k in range(V):
    for i in range(V):
        for j in range(V):
            if k==i or k==j: continue
            d[i][j] = min (d[i][j],
                d[i][k] + d[k][j])

not relax itself

TC = O(V³)
SC = O(V²) or O(2) [given array only]

If for any i d[i][i] < 0
There is a negative cycle →
This means for a node, a path to itself is always -ve. But if it is traversed in a negative cycle as our Int, then the -ve cycle will keep decreasing it → d[i][i].

Q Given a chessboard N×N
Given K knights placed in it.
If a knight is reachable from other knight in 1 step → you may swap them.
Find the no of ways of arranging Knight

→

Knight can move in these directions.

Since none of Knights can swap within reachable dist, there can be many arrangements possible.

In one connected component, a node can replace any node to form a diff arrangements

— — —

Any possible → K!
placement

Ans = K! × 2!

→ Ans = 1
Find CC of Knights
Find no of nodes in each CC
Ans = Ans × (# nodes)!

DFS on each cell in the matrix
Check if K and if it is reachable from current K.

Size of chessboard = N×N

TC → V → N×N
E → N(N-1)

TC = O(V+E)
= O(N²)

SC = O(V) = O(N²)
Size we should need allotted array