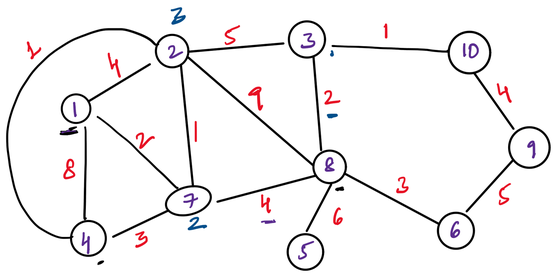
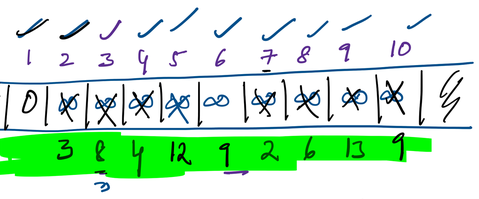
Graphs 5

Wednesday, 28 July 2021 4:56 PM

Single Source Shortest Paths

Dijkstra's + ve weights





Repeat prev till all node marked = N

1) Get unmarked node with minval (u) } O(N)

Visit its neighbours

for(int v : g[u]){

  dist[v] = min(dist[v], dist[u] + u→v)

}

overall TC:

→ O(N²+E)

Adj list

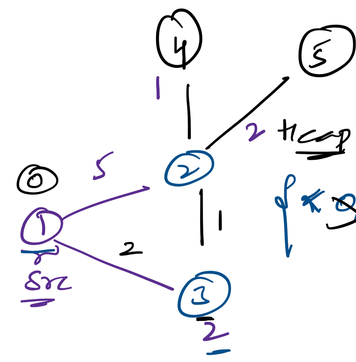
u	v	w
1	(2,4)	(4,8) (7,2)
2	(1,4)	(3,5), (7,1), (8,9)
3	(2,5)	(8,2) (10,1)
4	(1,8)	(2,1) (7,3)
5	(8,6)	
6	(8,3)	(9,5)
7	(1,2)	(2,1) (4,3) (8,4)
8	(2,9)	(3,2) (5,6) (6,8) (7,4)
9	(6,5)	(10,4)
10	(3,1)	(9,4)

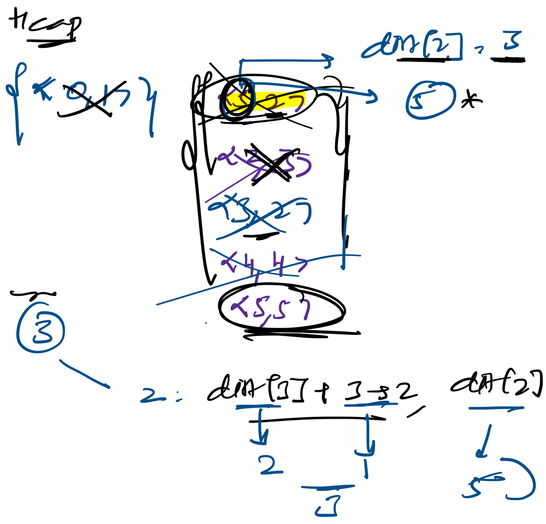
Use minheap to optimise find node with minimum weight

(We push both weight & Node in Heap)

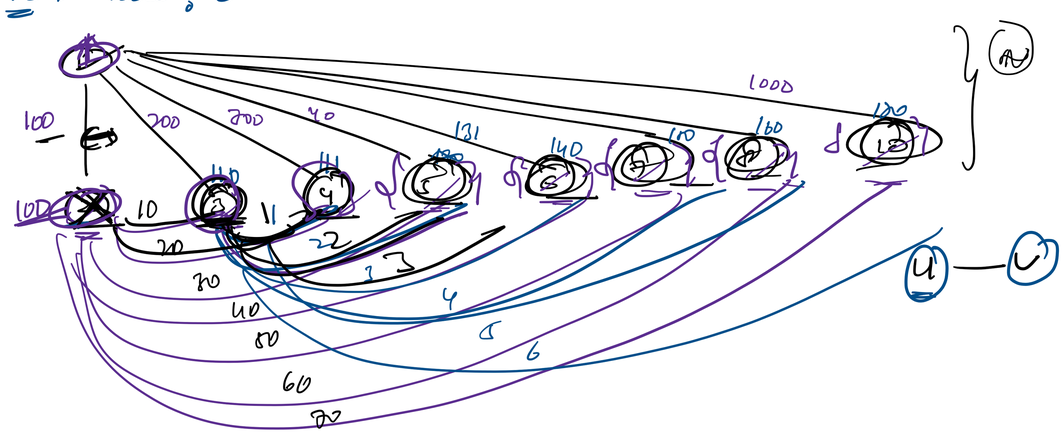
```
int d[N+1] = {inf}; dis[s] = 0;
minheap < pair<int, int> > hm; hm.insert({0, src})
while( hm.sizec()>0 ){
    pair< int, int > data = hm.top(); hm.delete, delet top
    int u = data.second
    int d = data.first; }
    if( d > dist[u] ) { continue }
    for( auto x : graph[u] ){
        int v = x.first, int w = x.second
        if( dist[v] > dist[u] + w ) {
            dist[v] = dist[u]+w // update
            hm.insert( { dist[v], v } );
        }
    }
}
```

Same node inserting multiple times in heaps





TC : Todo?

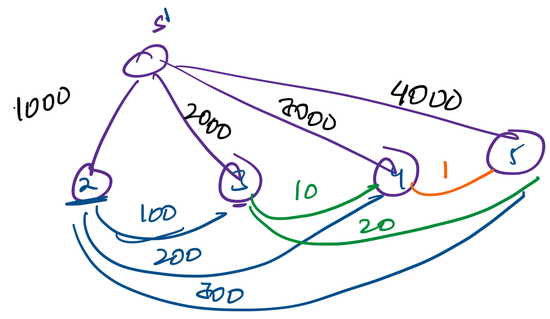


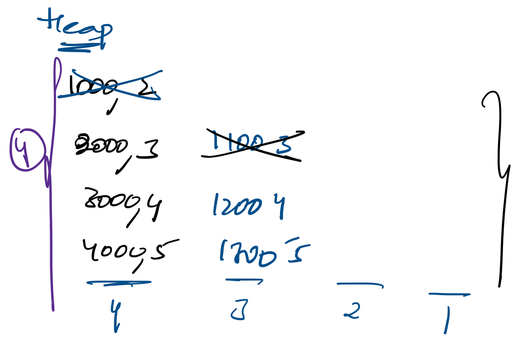
Edge { 1) for every Edge can we do an update

2) For every update we will push inside heaps }

3) At max how many time we push Todo

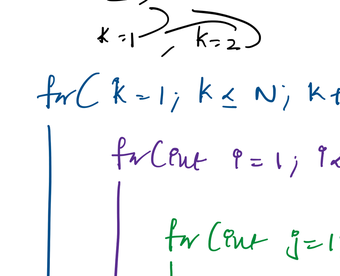
→ TC : ElogE SC : O(E)



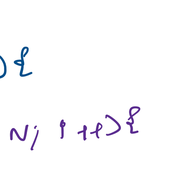


Q) Why Dijkstra won't work for negative Edges.

Eg:



A → C → C → B



// All pair shortest path

dp[i][j][k]: Get shortest path from (i → j) Inbetween node 0 to k

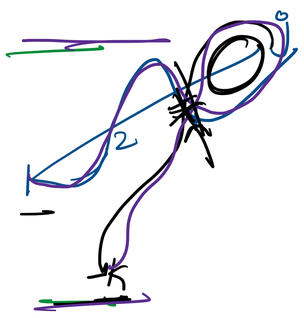
(i-j) can only be from (i → k)

{ k=0 } no intermediate node between i→j — if Edge i-j: dp[i][j]=w; if No Edg: dp[i][j] = ∞

k=1, k=2

```
for( k=1; k<=N; k++){
    for(int i=1; i<=N; i++){
        for(int j=1; j<=N; j++){
            if( dp[i][k] == ∞ || dp[k][j] == ∞) {continue}
            dp[i][j][k] = min( dp[i][j] , dp[i][k] + dp[k][j] )
        }
    }
}
```

TC: O(N³) SC: O(N³) → 2 Matrix only → 1 Matrix

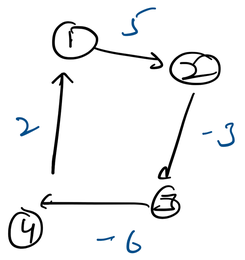


→ dp[i][k][k-1] = dp[i][k][k] (k)

→ dp[k][j][k-1] = dp[k][j][k]

{ k-j } { i, k-1 } [k-j], if i, -k

TC: O(N³) SC: O(N³) → Works for negative Edges



Initially

dp[1][1] = 0

dp[2][2] = 0

dp[N][N] = 0

After All Iterat.

N

∀ i=1 if(dp[i][i] < 0)

Negative cycle