G is a directed acyclic graph. You want to move from vertex **c** to vertex **z.** Some edges reduce your profit and some increase your profit. How do you get from c to z **while maximizing your profit**. What is the time complexity?

The time complexity: topological sort (TS) takes `O(V+E)`. In the worst case where `z` is a leaf and all other vertices point to it, we will visit all the graph edges which gives `O(V+E)`.
weight(u,v) is the cost of the edge (u,v)

G is a DAG with negative edges.

> Some edges reduce your profit and some increase your profit

- Edges - increase profit - positive value
- Edges - decrease profit - negative value

After TS, for each vertex U in TS order - relax each outgoing edge.

```
dist[] = {-INF, -INF, ….}
dist[c] = 0 // source

for every vertex u in topological order
  if (u == z) break; // dest vertex
  for every adjacent vertex v of u
    if (dist[v] < (dist[u] + weight(u, v))) // < for longest path = max profit
      dist[v] = dist[u] + weight(u, v)

ans = dist[z];
```