

# Shortest Path + Dijkstra

- weighted graph: each edge has weight  $w(e)$ 
  - distance
  - friendship/connection
  - connection costs

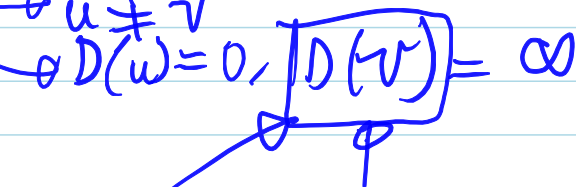
- $G$  is weighted graph  $(\therefore)$  length  $w(p)$  of path  $p = \sum_{i=1} w(\text{edge } i)$ 
  - NON-negative weights!  
otherwise -ve cycles

① Problem: Find SHORTEST path from  $A$  to  $Z$   
→ no -ve edges: Dijkstra's algorithm

- Dijkstra:
  - cloud of nodes, each at <sup>increasing distance</sup> distance away from  $n_i$ 
    - nodes enter based on I.D. (PP)
  - Terminates when no nodes left in cloud

- edge relaxation: replacing initially  $\infty$  values from  $u$  to  $v$  with actual values

→  $u \neq v$   
→  $D(u) = 0, D(v) = \infty$



Heap

Priority Queue contains all these

$O(\log n)$  removed  $\smile$  for ONE vertex

Up/Down heap bubbling

→ when inserting ALL nodes,  $O(n \log n)$

$$\rightarrow \sum_{u \in G} (1 + \deg(u)) \cdot \log(n) = O((n + e) \cdot \log(n))$$

→ overall runtime of while loop (check the pseudocode)

→ Bellman Ford: finds shortest paths in DIGRAPHS with -ve edge weights

→ main loop:  $n-1$  times

→ during loop: all edges visited

→  $O(1)$  operations

∴ Complexity:  $O(n \cdot e)$

→ Dijkstra > better for < sparse > graphs  
BF < dense >

