Single Source Shortest Paths

Dijkstra's Algorithm

Single-Source Shortest Paths

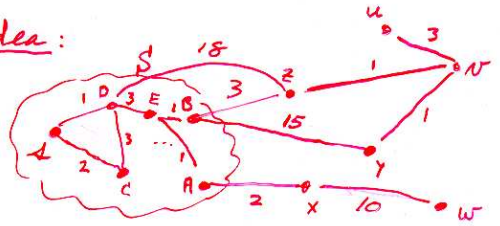
$G = (V, E)$

$w: E \to \mathbb{R}$ edge weights/lengths

$s \in V$ source

want: length of shortest path $s \to v$ $\forall v$

Idea:



Single-Source Shortest Paths

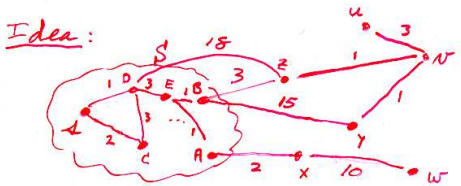
$G = (V, E)$

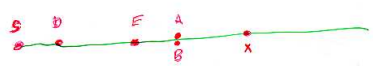
$w : E \to \mathbb{R}$ edge weights/lengths

$s \in V$ Source

want: length of shortest path $s \to v$ $\forall v$

Idea:





Vertex	d			
s	0			
A	5	In S		
B	5			
C	2			
D	1			
E	4			
u	~~∞~~		12 add to S	add
v	~~∞~~	9		add
w	~~∞~~ 7+w=17			
X	7 add to S		10 add to S	
Y	~~20~~			
Z	8 add to S			

Dijkstra's Algorithm

Initialize

$\forall v \quad d(v) \leftarrow w(s, v)$

NB: $d(s) = 0$

$d(v) = \infty$ if not adjacent to s

$S \leftarrow \{s\}$

$Q \leftarrow$ all other v, keyed by $d(v)$

While Q not empty

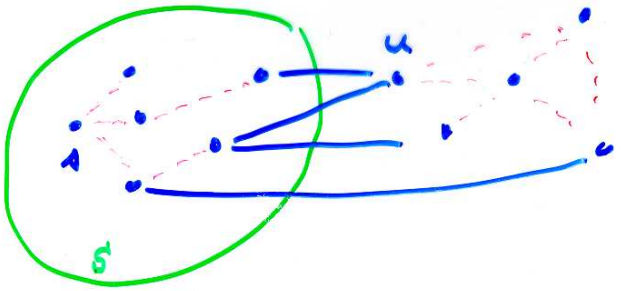
$u \leftarrow \text{EXTRACTMIN}(Q)$

$S \leftarrow S \cup \{u\}$

$\forall$ edges (u, v)

$d(v) \leftarrow \min(d(v), d(u) + w(u,v))$

Correctness



Assume $w(e) > 0$ for all edges e

Suppose S contains closest $|S|$ nodes

Let u be closest node not in S

Let $s, v_1, v_2, \ldots, v_k, u$ be a shortest path to u. Then $v_1 \ldots v_k$ all $\in S$ (all closer than u, & u next closest.)

$\therefore$ Edge (v_k, u) among those considered when calculating $d(u)$.

For all u', $d(u') \geq$ actual min

$\therefore$ The alg will (correctly) pick u next.

Dijkstra's Algorithm

Implementation & Analysis

Key issue: priority queue

1. Search array

   O(n) per MIN, n^2 total
   O(1) per update, e total
   } $O(n^2)$

2. Heap

   O(log n) per min, n log n total
   O(log n) per update, e log n total
) $(n+e)\log n$

3. Fibonacci Heap

   $n \log n + e$