

Shortest Paths I: Properties, Dijkstra's Algorithm

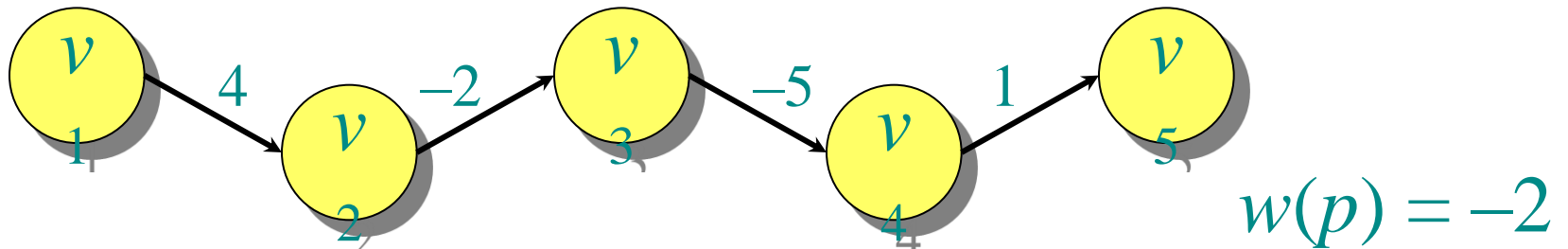
Lecture 14

Paths in graphs

Consider a digraph $G = (V, E)$ with edge-weight function $w : E \rightarrow \mathbb{R}$. The **weight** of path $p = v_1 \rightarrow v_2 \rightarrow \dots \rightarrow v_k$ is defined to be

$$w(p) = \sum_{i=1}^{k-1} w(v_i, v_{i+1}).$$

Example:



Shortest paths

A *shortest path* from u to v is a path of minimum weight from u to v . The *shortest-path weight* from u to v is defined as

$$\delta(u, v) = \min\{w(p) : p \text{ is a path from } u \text{ to } v\}.$$

Note: $\delta(u, v) = \infty$ if no path from u to v exists.

Single-source shortest paths

Problem. From a given source vertex $s \in V$, find the shortest-path weights $\delta(s, v)$ for all $v \in V$.

If all edge weights $w(u, v)$ are *nonnegative*, all shortest-path weights must exist.

IDEA: Greedy.

1. Maintain a set S of vertices whose shortest-path distances from s are known.
2. At each step add to S the vertex $v \in V - S$ whose distance estimate from s is minimal.
3. Update the distance estimates of vertices adjacent to v .

Dijkstra's algorithm

$d[s] \leftarrow 0$

for each $v \in V - \{s\}$

do $d[v] \leftarrow \infty$

$S \leftarrow \emptyset$

$Q \leftarrow V$ \triangleright Q is a priority queue maintaining $V - S$

while $Q \neq \emptyset$

do $u \leftarrow \text{EXTRACT-MIN}(Q)$

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

for each $v \in \text{Adj}[u]$

do **if** $d[v] > d[u] + w(u, v)$

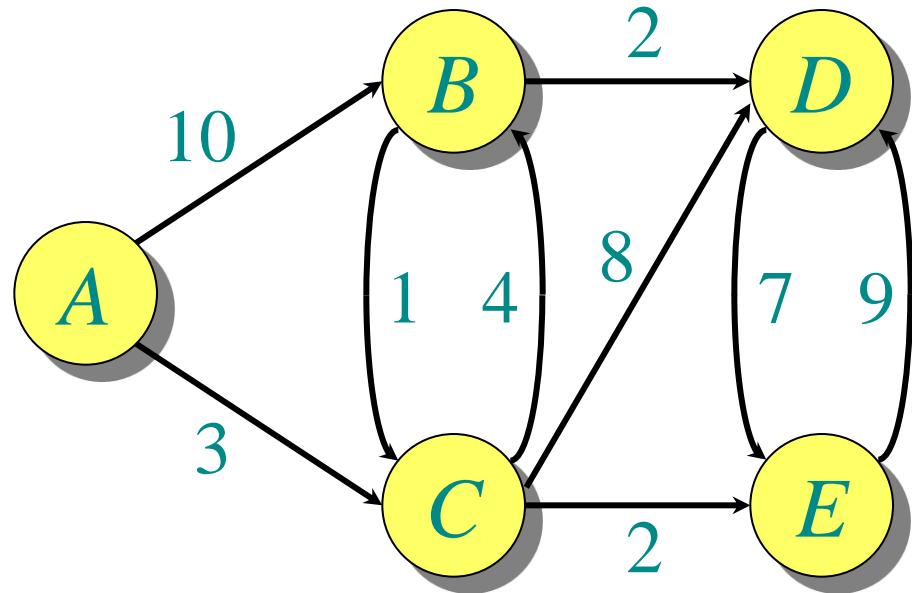
then $d[v] \leftarrow d[u] + w(u, v)$

*relaxation
step*

Implicit  DECREASE-KEY

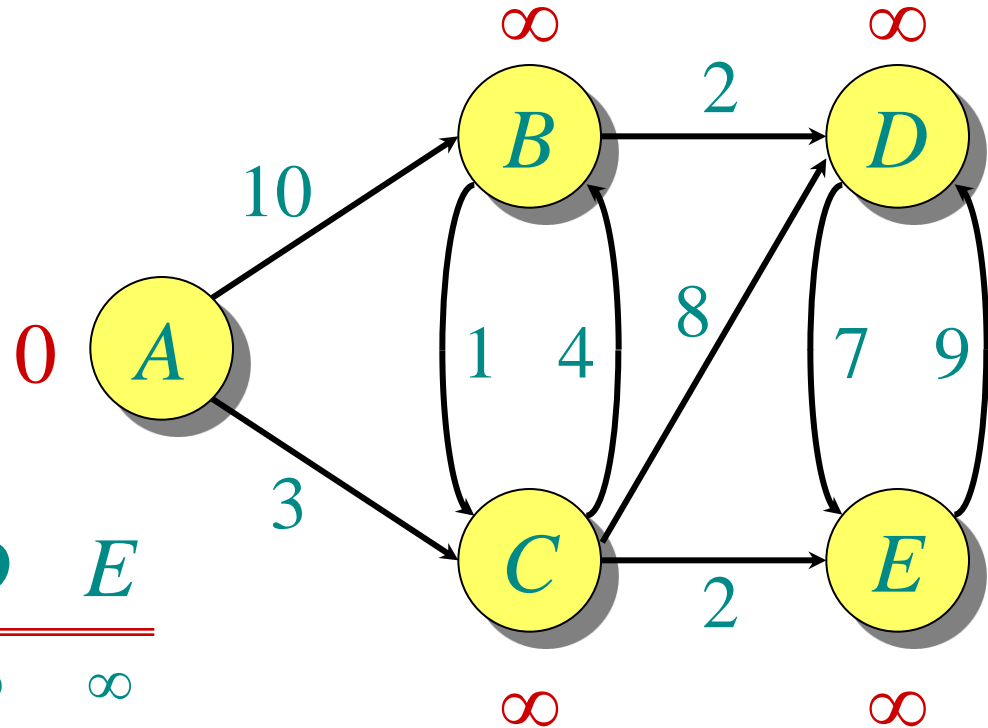
Example of Dijkstra's algorithm

Graph with nonnegative edge weights:



Example of Dijkstra's algorithm

Initialize:



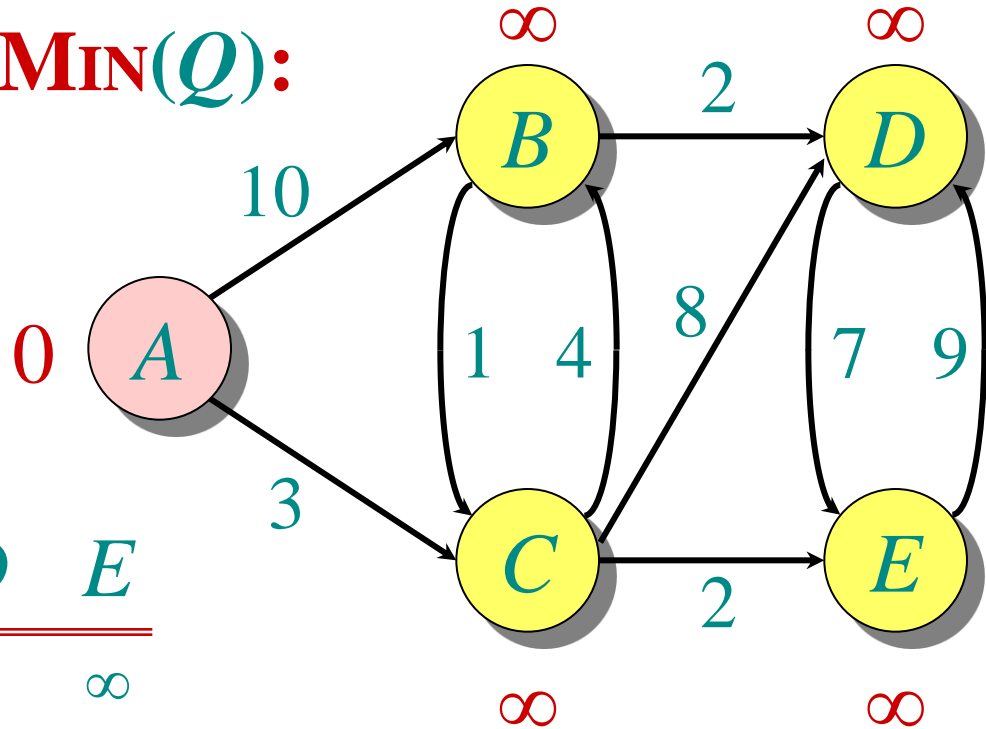
Q:

<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	<i>E</i>
0	∞	∞	∞	∞

S: {}

Example of Dijkstra's algorithm

“A” ← **EXTRACT-MIN**(*Q*):



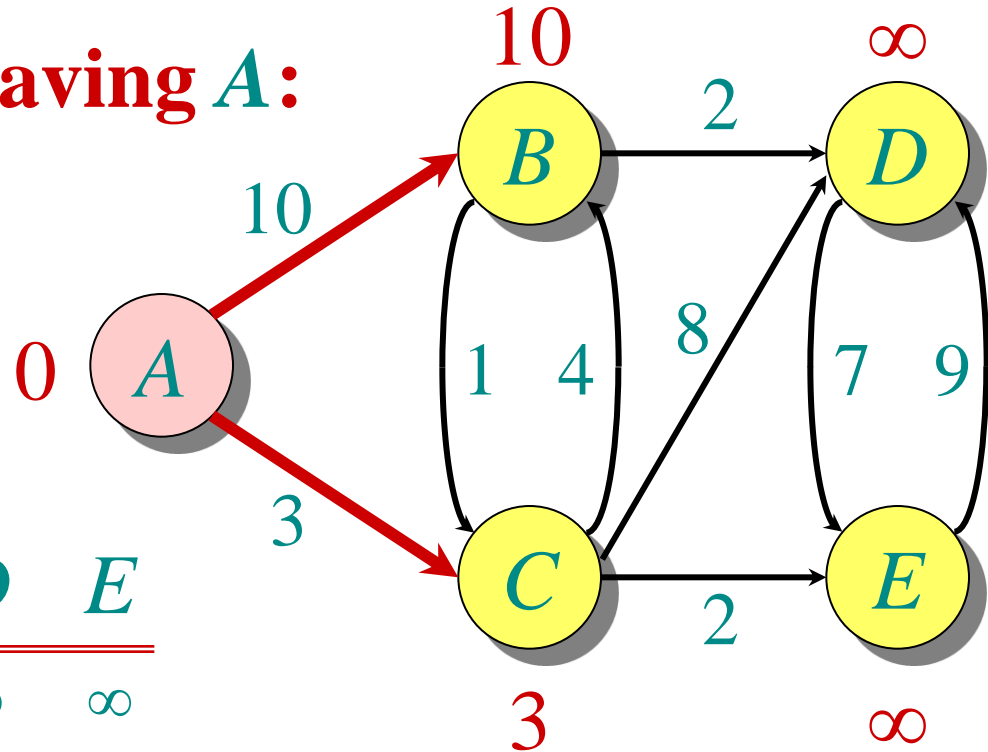
Q:

<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	<i>E</i>
0	∞	∞	∞	∞

S: { *A* }

Example of Dijkstra's algorithm

Relax all edges leaving *A*:



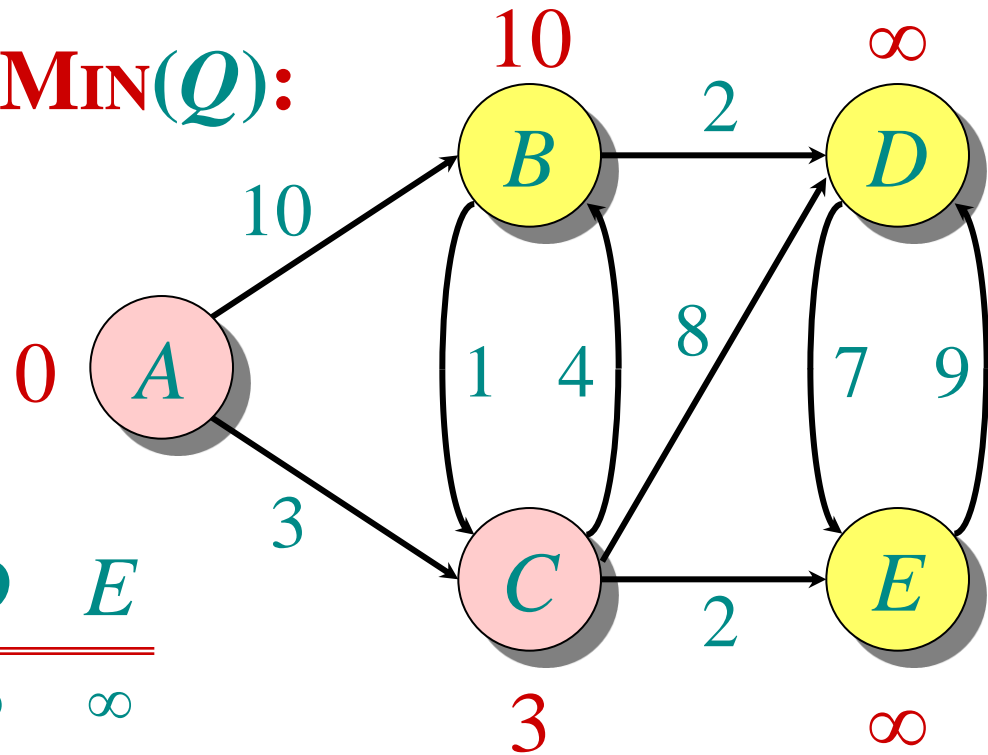
Q:

<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	<i>E</i>
0	∞	∞	∞	∞
	10	3	-	-

S: { *A* }

Example of Dijkstra's algorithm

“C” ← **EXTRACT-MIN**(Q):



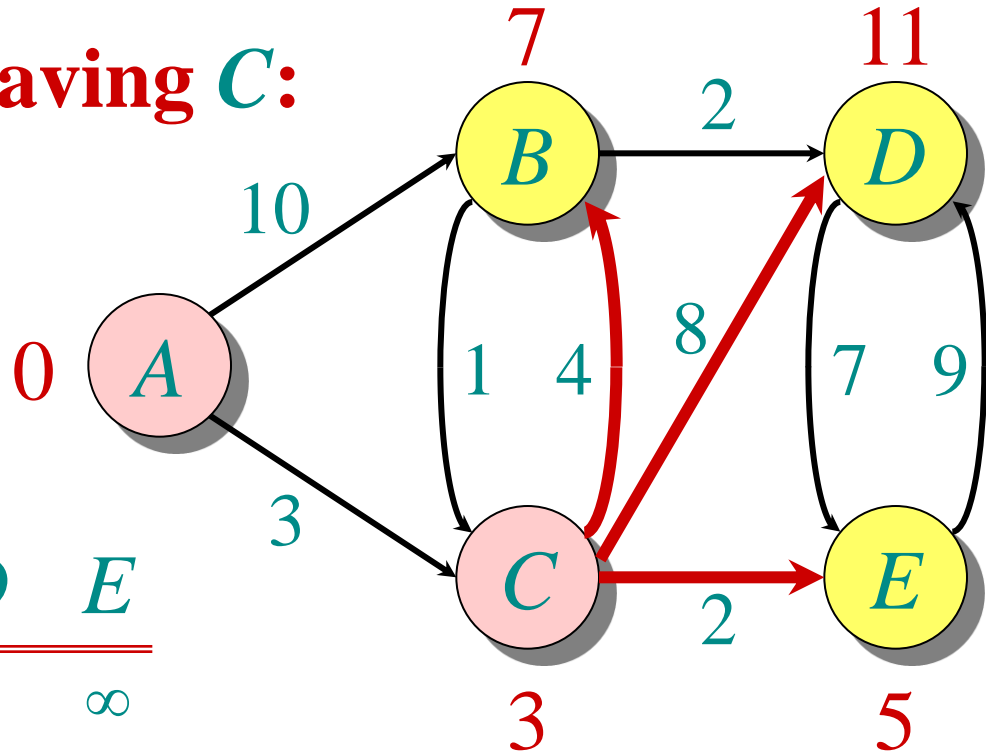
Q:

A	B	C	D	E
0	∞	∞	∞	∞
	10	3	-	-

S: { A, C }

Example of Dijkstra's algorithm

Relax all edges leaving C :



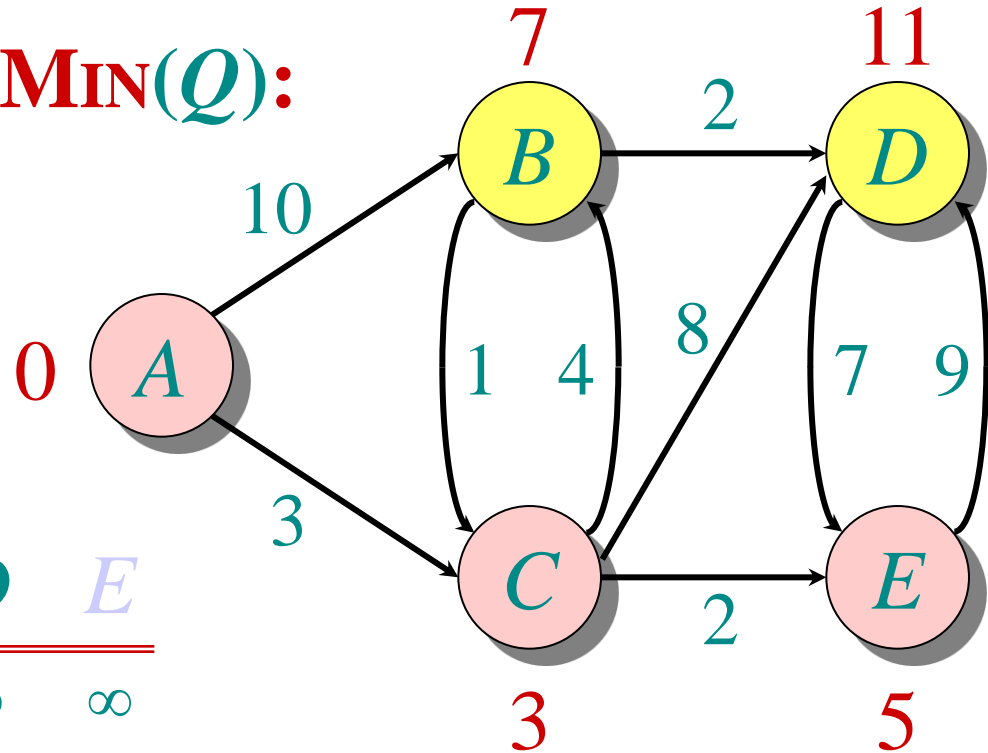
Q :

A	B	C	D	E
0	∞	∞	∞	∞
	10	3	—	—
	7		11	5

$S: \{ A, C \}$

Example of Dijkstra's algorithm

“E” ← **EXTRACT-MIN(Q)**:



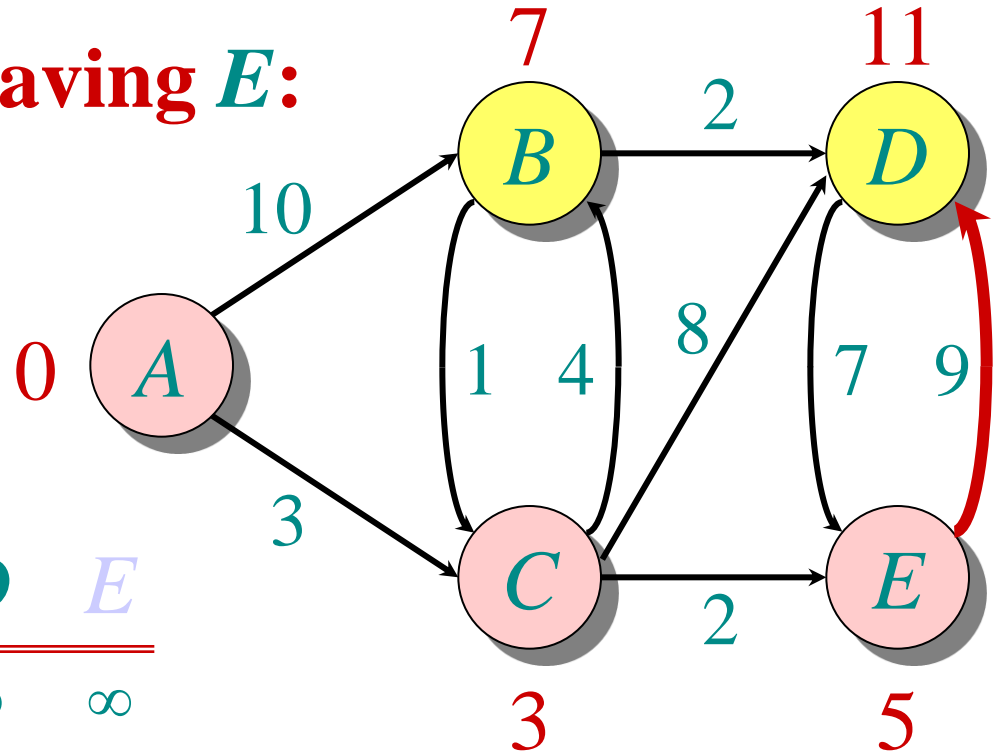
Q:

A	B	C	D	E
0	∞	∞	∞	∞
	10	3	-	-
	7		11	5

S: { A, C, E }

Example of Dijkstra's algorithm

Relax all edges leaving E :



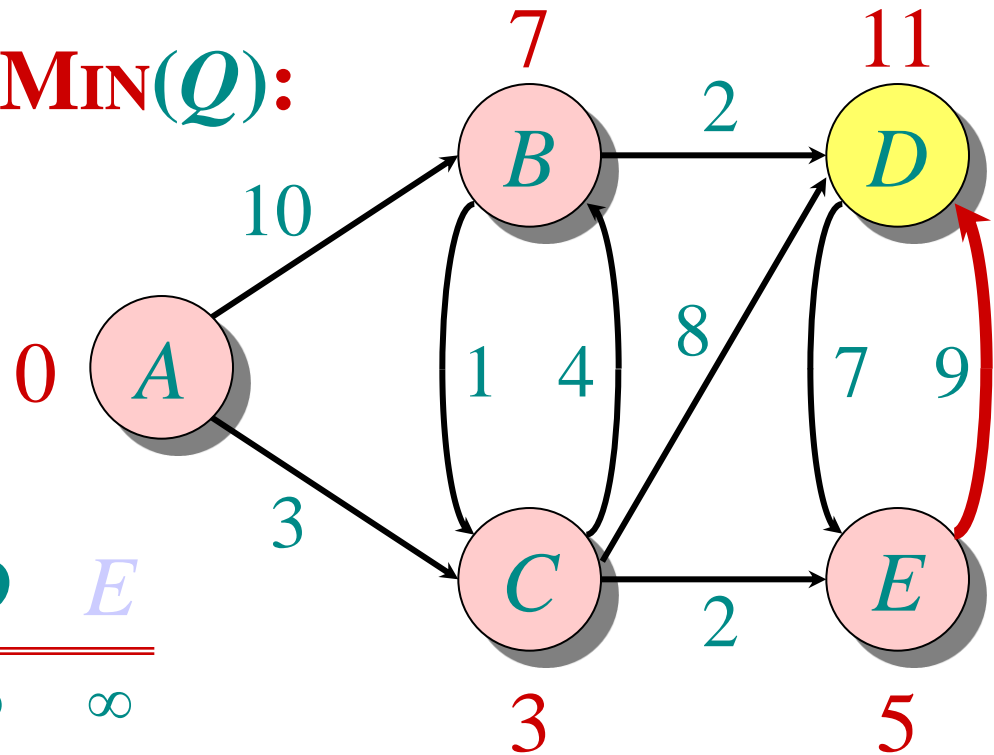
Q :

A	B	C	D	E
0	∞	∞	∞	∞
	10	3	∞	∞
	7		11	5
	7		11	

$S: \{ A, C, E \}$

Example of Dijkstra's algorithm

“B” ← **EXTRACT-MIN(Q)**:



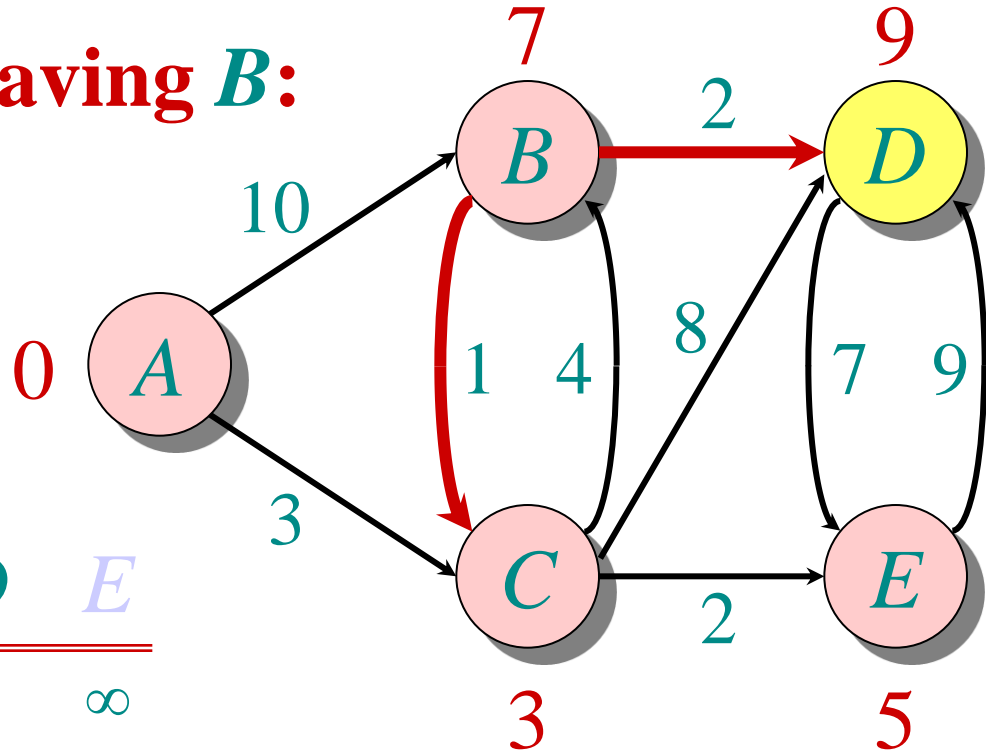
Q:

A	B	C	D	E
0	∞	∞	∞	∞
	10	3	∞	∞
	7		11	5
	7		11	

S: { A, C, E, B }

Example of Dijkstra's algorithm

Relax all edges leaving *B*:



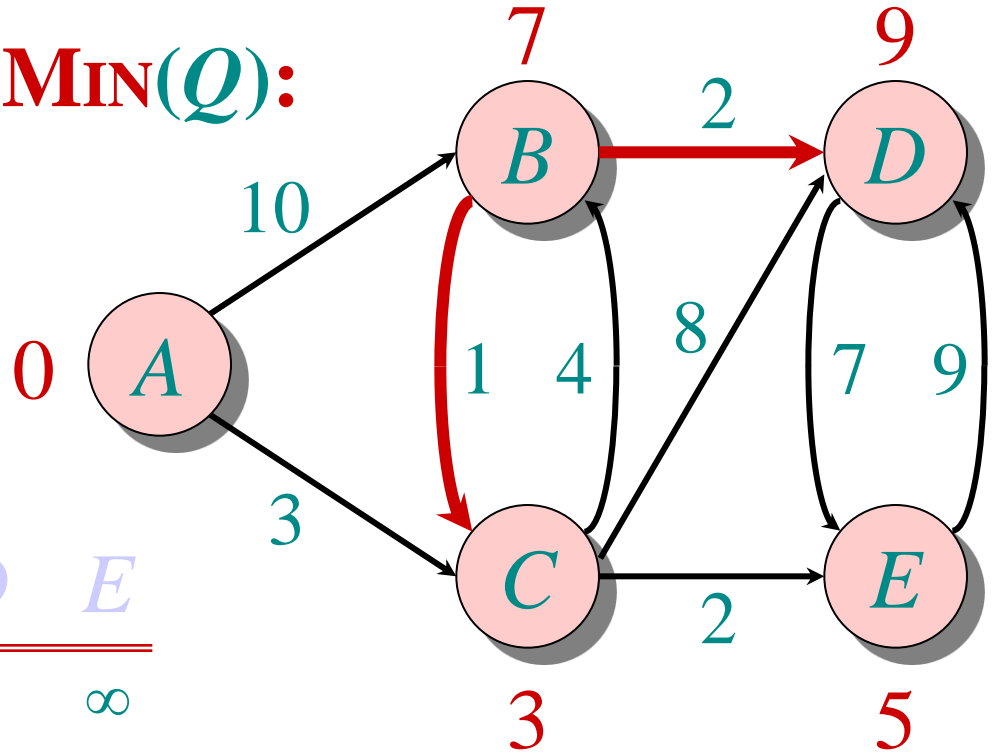
Q:

<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	<i>E</i>
0	∞	∞	∞	∞
	10	3	∞	∞
	7		11	5
	7		11	
			9	

S: { *A*, *C*, *E*, *B* }

Example of Dijkstra's algorithm

“D” ← **EXTRACT-MIN(Q)**:



Q:

A	B	C	D	E
0	∞	∞	∞	∞
	10	3	∞	∞
	7		11	5
	7		11	
			9	

S: { A, C, E, B, D }

Analysis of Dijkstra

$|V|$ times { while $Q \neq \emptyset$
do $u \leftarrow \text{EXTRACT-MIN}(Q)$
 $S \leftarrow S \cup \{u\}$
for each $v \in \text{Adj}[u]$
do if $d[v] > d[u] + w(u, v)$
then $d[v] \leftarrow d[u] + w(u, v)$

$\text{degree}(u)$ times {

Handshaking Lemma $\Rightarrow \Theta(E)$ implicit DECREASE-KEY's.

Time = $\Theta(V) \cdot T_{\text{EXTRACT-MIN}} + \Theta(E) \cdot T_{\text{DECREASE-KEY}}$

Note: Same formula as in the analysis of Prim's minimum spanning tree algorithm.

Analysis of Dijkstra (continued)

$$\text{Time} = \Theta(V) \cdot T_{\text{EXTRACT-MIN}} + \Theta(E) \cdot T_{\text{DECREASE-KEY}}$$

Q	$T_{\text{EXTRACT-MIN}}$	$T_{\text{DECREASE-KEY}}$	Total
array	$O(V)$	$O(1)$	$O(V^2)$
binary heap	$O(\lg V)$	$O(\lg V)$	$O(E \lg V)$
Fibonacci heap	$O(\lg V)$ amortized	$O(1)$ amortized	$O(E + V \lg V)$ worst case