

CS 445

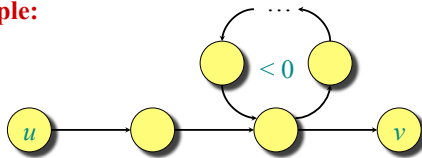
Shortest Paths in Graphs Bellman-Ford Algorithm

Slides courtesy of Erik Demaine and
Carola Wenk

Negative-weight cycles

Recall: If a graph $G = (V, E)$ contains a negative-weight cycle, then some shortest paths may not exist.

Example:



Bellman-Ford algorithm: Finds all shortest-path lengths from a *source* $s \in V$ to all $v \in V$ or determines that a negative-weight cycle exists.

Bellman-Ford and Undirected graphs

Bellman-Ford algorithm is designed for **directed** graphs.

If G is undirected, replace every edge (u, v) with two directed edges (u, v) and (v, u) , both with weight $w(u, v)$

Bellman-Ford algorithm

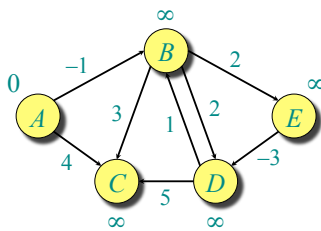
```

 $d[s] \leftarrow 0$ 
for each  $v \in V - \{s\}$  do  $d[v] \leftarrow \infty$  } initialization
for  $i \leftarrow 1$  to  $|V| - 1$  do
  for each edge  $(u, v) \in E$  do
    if  $d[v] > d[u] + w(u, v)$  then } relaxation
       $d[v] \leftarrow d[u] + w(u, v)$  } step
       $\pi[v] \leftarrow u$ 
for each edge  $(u, v) \in E$ 
  do if  $d[v] > d[u] + w(u, v)$ 
    then report that a negative-weight cycle exists
  
```

At the end, $d[v] = \delta(s, v)$. Time = $O(|V| |E|)$.

Example of Bellman-Ford

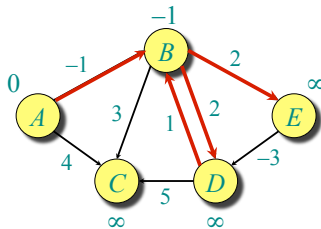
Order of edges: $(B,E), (D,B), (B,D), (A,B), (A,C), (D,C), (B,C), (E,D)$



A	B	C	D	E
0	∞	∞	∞	∞

Example of Bellman-Ford

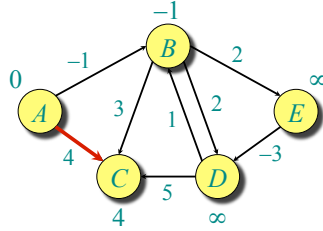
Order of edges: $(B,E), (D,B), (B,D), (A,B), (A,C), (D,C), (B,C), (E,D)$



A	B	C	D	E
0	∞	∞	∞	∞
0	-1	∞	∞	∞

Example of Bellman-Ford

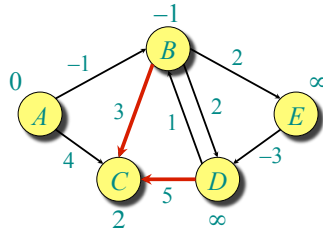
Order of edges: $(B,E), (D,B), (B,D), (A,B), (A,C), (D,C), (B,C), (E,D)$



A	B	C	D	E
0	∞	∞	∞	∞
0	-1	∞	∞	∞
0	-1	4	∞	∞

Example of Bellman-Ford

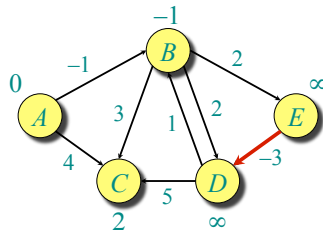
Order of edges: $(B,E), (D,B), (B,D), (A,B), (A,C), (D,C), (B,C), (E,D)$



A	B	C	D	E
0	∞	∞	∞	∞
0	-1	∞	∞	∞
0	-1	4	∞	∞
0	-1	2	∞	∞

Example of Bellman-Ford

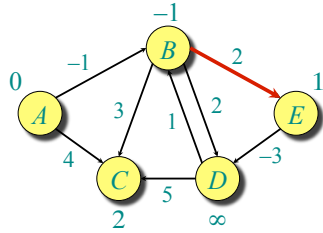
Order of edges: $(B,E), (D,B), (B,D), (A,B), (A,C), (D,C), (B,C), (E,D)$



A	B	C	D	E
0	∞	∞	∞	∞
0	-1	∞	∞	∞
0	-1	4	∞	∞
0	-1	2	∞	∞

Example of Bellman-Ford

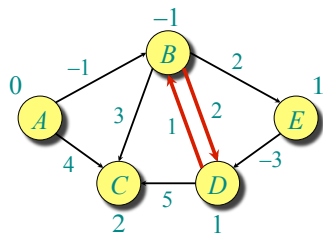
Order of edges: $(B,E), (D,B), (B,D), (A,B), (A,C), (D,C), (B,C), (E,D)$



A	B	C	D	E
0	∞	∞	∞	∞
0	-1	∞	∞	∞
0	-1	4	∞	∞
0	-1	2	∞	∞
0	-1	2	∞	1

Example of Bellman-Ford

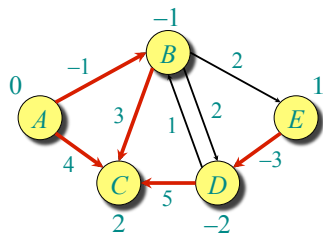
Order of edges: $(B,E), (D,B), (B,D), (A,B), (A,C), (D,C), (B,C), (E,D)$



A	B	C	D	E
0	∞	∞	∞	∞
0	-1	∞	∞	∞
0	-1	4	∞	∞
0	-1	2	∞	∞
0	-1	2	∞	1
0	-1	2	1	1

Example of Bellman-Ford

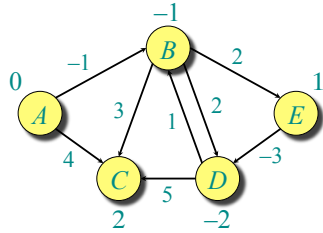
Order of edges: $(B,E), (D,B), (B,D), (A,B), (A,C), (D,C), (B,C), (E,D)$



A	B	C	D	E
0	∞	∞	∞	∞
0	-1	∞	∞	∞
0	-1	4	∞	∞
0	-1	2	∞	∞
0	-1	2	∞	1
0	-1	2	1	1
0	-1	2	-2	1

Example of Bellman-Ford

Order of edges: $(B,E), (D,B), (B,D), (A,B), (A,C), (D,C), (B,C), (E,D)$



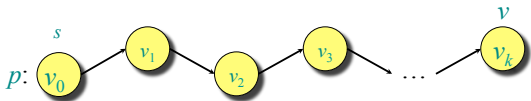
	A	B	C	D	E
0	∞	∞	∞	∞	∞
0	-1	∞	∞	∞	∞
0	-1	4	∞	∞	∞
0	-1	2	∞	∞	∞
0	-1	2	∞	1	1
0	-1	2	1	1	1
0	-1	2	-2	1	1

Note: Values decrease monotonically.

Correctness

Theorem. If $G = (V, E)$ contains no negative-weight cycles, then after the Bellman-Ford algorithm executes, $d[v] = \delta(s, v)$ for all $v \in V$.

Proof. Let $v \in V$ be any vertex, and consider a shortest path p from s to v with the minimum number of edges.



Since p is a shortest path, we have

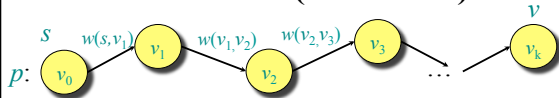
$$\delta(s, v_i) = \delta(s, v_{i-1}) + w(v_{i-1}, v_i) \text{ for every } i.$$

Correctness

Lemma If $G = (V, E)$ contains no negative-weight cycles, then during BF-algorithm $d[v] \geq \delta(s, v)$, for every vertex v , for all $v \in V$.

Proof: Same as in Dijkstra's algorithm.

Correctness (continued)



Let p be the shortest path from s to a vertex v . Lets re-label the vertices along p so $s=v_0, v_1, \dots, v_k=v$
 Note that (a portion of) p is also the shortest path to each v_i .

Initially, $d[v_0] = 0 = \delta(s, v_0)$, and $d[s]$ is unchanged by subsequent relaxations (note $\delta(s, s) \geq 0$ (why?)).

- After 1 pass through E , we have $d[v_1] = \delta(s, v_1) = w(s, v_1)$.
- After 2 passes through E , we have $d[v_2] = d[v_1] + w(v_1, v_2) = w(s, v_1) + w(v_1, v_2) = \delta(s, v_2)$.

...

- After k passes through E , we have $d[v_k] = \delta(s, v_k)$.

Since G contains no negative-weight cycles, p is simple. Longest simple path has $\leq |V| - 1$ edges. \square

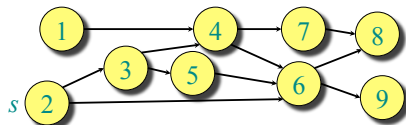
Detection of negative-weight cycles

Corollary. If a value $d[v]$ fails to converge after $|V| - 1$ passes, there exists a negative-weight cycle in G reachable from s . \square

DAG shortest paths

If the graph is a **directed acyclic graph (DAG)**, we first **topologically sort** the vertices.

- Determine $f: V \rightarrow \{1, 2, \dots, |V|\}$ such that $(u, v) \in E \Rightarrow f(u) < f(v)$.
- $O(V + E)$ time using depth-first search.



Walk through the vertices $u \in V$ in this order, relaxing the edges in $Adj[u]$, thereby obtaining the shortest paths from s in a total of $O(V + E)$ time.
