Algorithms

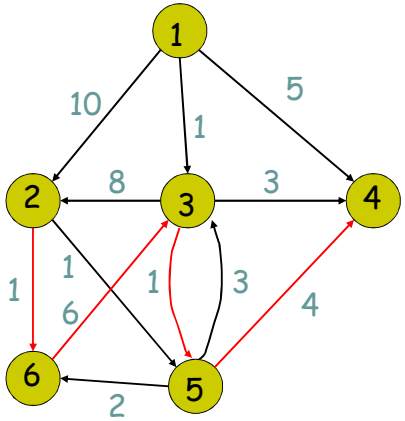
Dana Shapira

Lesson #7:

Single source shortest path problem

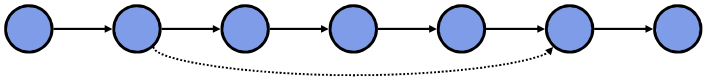
Single Source Shortest Path Problem

- Given a weighted, directed graph $G = (V, E)$, with weight function w: $E \rightarrow$ R mapping edges to real valued weights.
- **Single source shortest path problem**: given vertex s, for every vertex $v \in V$ find a shortest path from s to v.
- **Dijkstra's algorithm** solves this problem efficiently for the case in which all weights are nonnegative



Shortest Path Properties

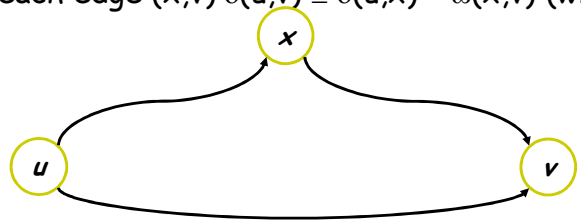
- We have *optimal substructure*: the shortest path consists of shortest subpaths:



- Proof: suppose some subpath is not a shortest path
  - There must then exist a shorter subpath
  - Could substitute the shorter subpath for a shorter path
  - But then overall path is not shortest path. Contradiction

Shortest Path Properties

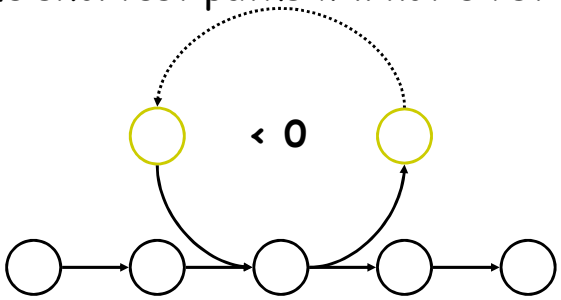
- Define $\delta(u,v)$ to be the weight of the shortest path from u to v
- Shortest paths satisfy the *triangle inequality*:
  $\delta(u,v) \leq \delta(u,x) + \delta(x,v)$
- For each edge (x,v) $\delta(u,v) \leq \delta(u,x) + \omega(x,v)$ (why?)



This path is no longer than any other path

Shortest Path Properties

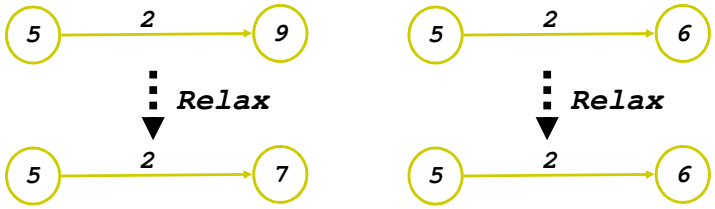
- In graphs with negative weight cycles, some shortest paths will not exist (*Why?*):



Relaxation

For all *v*, maintain upper bound d[*v*] on $\delta(s,v)$

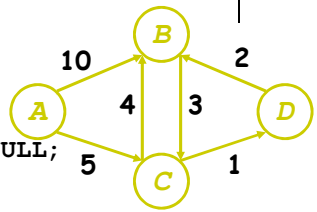
```
Relax(u,v,w) {
  if (d[v] > d[u]+w(u,v)) then
      d[v]=d[u]+w(u,v);
}
```



Dijkstra's Algorithm

```
Dijkstra(G,s)
   for each v ∈ V
      d[v] = ∞; π(v)=NULL;
   d[s] = 0; S = ∅; Q = V; π(s)=NULL;
   while (Q ≠ ∅)
      u = ExtractMin(Q);
      S = S U {u};
      for each v ∈ Adj[u]
         if (d[v] > d[u]+w(u,v))
            d[v] = d[u]+w(u,v);
            π(v)=u
```

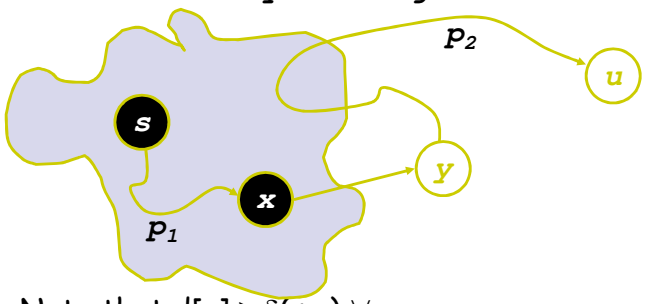
Relaxation Step



What will be the total running time?

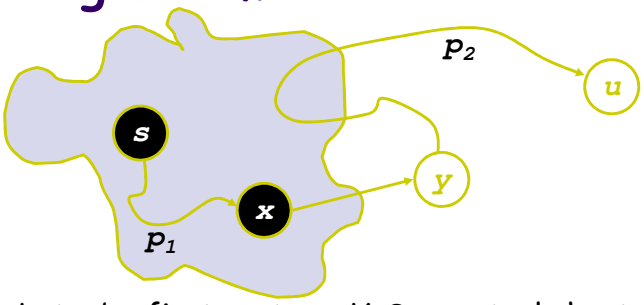
Correctness Of Dijkstra's Algorithm

We must show that when u is removed from Q, it has already converged



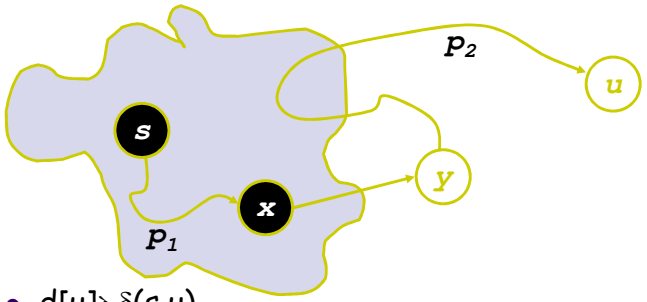
- Note that $d[v] \geq \delta(s,v)\ \forall v$
- Let u be first vertex in S such that $d[u] \neq \delta(s,u)$ when it is removed from Q.

Correctness Of Dijkstra's Algorithm



- Let y be first vertex $\in$V-S on actual shortest path from s→u $\Rightarrow$ d[y] = δ(s,y)
  - Because d[x] is set correctly for y's predecessor x $\in$ S on the shortest path (u was the first vertex in S such that d[u] $\neq$ δ(s,u)) ,
  - When we put x into S, we relaxed (x,y), giving d[y] the correct value

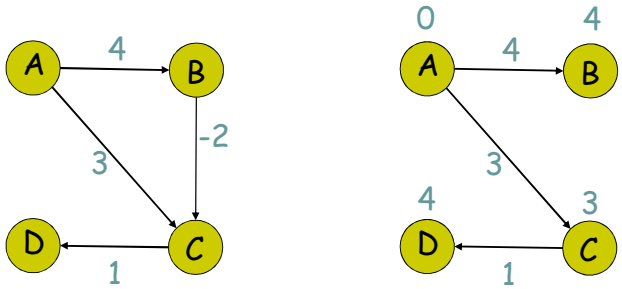
Correctness Of Dijkstra's Algorithm



- d[u]> δ(s,u)
  = δ(s,y) + δ(y,u) (*Why?*)
  = d[y] + δ(y,u)
  $\geq$ d[y] But if d[u] > d[y], wouldn't have chosen u. Contradiction.

Question

- Can you give an example of a graph that includes negative weights, and does not work with Dijkstra's algorithm?



Bellman-Ford Algorithm

```
BellmanFord()
   for each v ∈ V
      d[v] = ∞;
   d[s] = 0;
   for i=1 to |V|-1
      for each edge (u,v) ∈ E
         Relax(u,v, w(u,v));
   for each edge (u,v) ∈ E
      if (d[v] > d[u] + w(u,v))
            return "no solution";
```

Initialize d[], which will converge to shortest-path value δ

Relaxation: Make |V|-1 passes, relaxing each edge

Test for solution
Under what condition do we get a solution?

```
Relax(u,v,w): if (d[v] > d[u]+w) then d[v]=d[u]+w
```

Bellman-Ford Algorithm

```
BellmanFord()
   for each v ∈ V
      d[v] = ∞;
   d[s] = 0;
   for i=1 to |V|-1
      for each edge (u,v) ∈ E
         Relax(u,v, w(u,v));
   for each edge (u,v) ∈ E
      if (d[v] > d[u] + w(u,v))
           return "no solution";
```

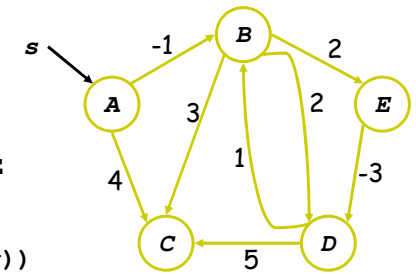
What will be the running time?

A: O(|V||E|)

```
Relax(u,v,w): if (d[v] > d[u]+w) then d[v]=d[u]+w
```


Bellman-Ford Algorithm

```
BellmanFord()
   for each v ∈ V
      d[v] = ∞;
   d[s] = 0;
   for i=1 to |V|-1
      for each edge (u,v) ∈ E
         Relax(u,v, w(u,v));
   for each edge (u,v) ∈ E
      if (d[v] > d[u] + w(u,v))
           return "no solution";
```



Example

```
Relax(u,v,w): if (d[v] > d[u]+w) then d[v]=d[u]+w
```

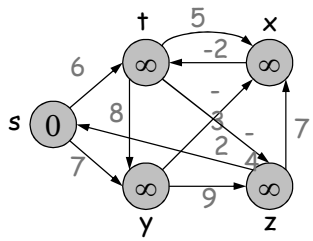

Bellman-Ford

- Correctness: show d[v] = δ(s,v) after |V|-1 passes
  - Lemma: d[v] $\geq$ δ(s,v) always
    - Initially true
    - Let v be first vertex for which d[v] < δ(s,v)
    - Let u be the vertex that caused d[v] to change: d[v] = d[u] + w(u,v)
    - Then d[v] < δ(s,v)
      $\delta(s,v) \leq \delta(s,u) + w(u,v)$ (*Why?*)
      $\delta(s,u) + w(u,v) \leq d[u] + w(u,v)$ (*Why?*)
    - So d[v] < d[u] + w(u,v). Contradiction.

Bellman-Ford

- Prove: after |V|-1 passes, all *d* values correct
  - Consider shortest path from s to v:
    $s \rightarrow v_1 \rightarrow v_2 \rightarrow v_3 \rightarrow \ldots \rightarrow v$
    - Initially, d[s] = 0 is correct, and doesn't change (*Why?*)
    - After 1 pass through edges, d[v_1] is correct (*Why?*) and doesn't change
    - After 2 passes, d[v_2] is correct and doesn't change
    - …
    - Terminates in |V| - 1 passes: (*Why?*)
    - *What if it doesn't?*

Bellman-Ford Example



DAG Shortest Paths

- Problem: finding shortest paths in DAG
  - Bellman-Ford takes O(|V||E|) time.
  - *How can we do better?*
  - Idea: use topological sort
    - If were lucky and processes vertices on each shortest path from left to right, would be done in one pass
    - Every path in a DAG is subsequence of topologically sorted vertex order, so processing vertices in that order, we will do each path in forward order (will never relax edges out of vertices before doing all edges into vertices).
    - Thus: just one pass. *What will be the running time?*