26 Feb 2021 — The Bellman-Ford Algorithm (§6.8)

Announcements

1. Extra office hour today 11-12 EST. (see course website for Prof Kleinberg's office hour link)
2. Problem Set 3 deadline will be Friday 3/5. (Not the usual Thursday.) Some TAs will move their office hours from 3/9-10 to 3/5 to allow for extra coverage.

Shortest Paths in a Directed Graph:

Given directed graph $G = (V, E)$ and edge cost function $cost: E \to \mathbb{R}$ (includes negative numbers.)

Given source vertex s, destination vertex t, find minimum cost ~~path~~ walk from s to t... if one exists.

"Path" means a sequence of vertices

$$s = u_0, u_1, u_2, \ldots, u_k = t$$

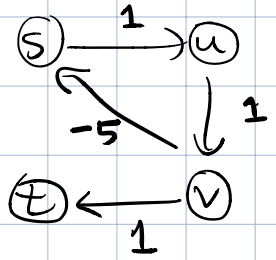
such that (u_i, u_{i+1}) is a directed edge of G for $i = 0, 1, \ldots, k-1$.

"Simple path" means no vertex is repeated.

"Walk" means vertices could be repeated.

The word "path" is ambiguous ... some people mean "simple path" and others mean "walk".

If there are no walks from s to t, a min-cost walk will not exist.



	Cost
s u v t	3
s u v s u v t	∅
s u v s u v s u v t	-3
⋮	⋮

A minimum cost walk from s to t exists if and only if:

- the set of s-t walks is non-empty
- none of them contains a cycle of strictly negative cost.

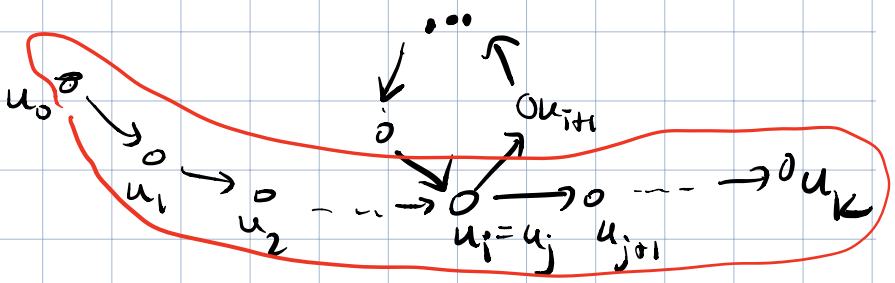
Furthermore, when these two conditions hold, at least one of the min-cost walks from s to t is a simple path.

↳ Proof. If W is a min-cost walk from s to t, assume without loss of

generality that W is one of the min-cost walks with the fewest vertices.

Now, if $W = u_0, u_1, \ldots, u_k$

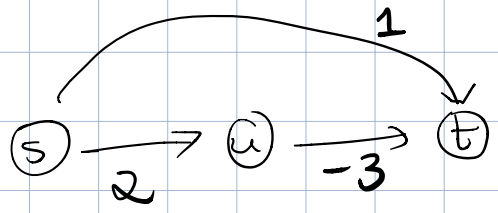
Suppose (by way of contradiction) W is not a simple path. Then $\exists\ i<j$ with $u_i = u_j$.



By assumption no walk from s to t contains a negative cost cycle.
So the walk

$$u_0 \to u_1 \to \cdots \to u_i \to u_{j+1} \to u_{j+2} \to \cdots \to u_k$$

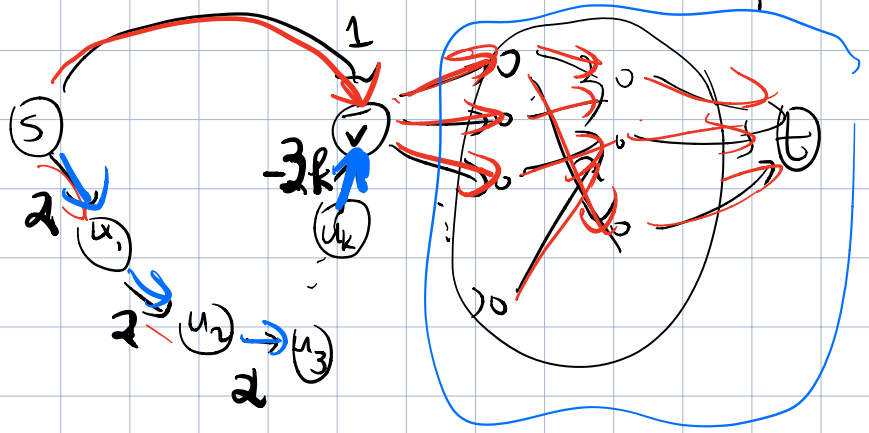
has strictly fewer vertices and no greater cost than W. ↯ (contradiction)



		Cost
Direct:	$s \to t$	1
Indirect:	$s \to u \to t$	-1

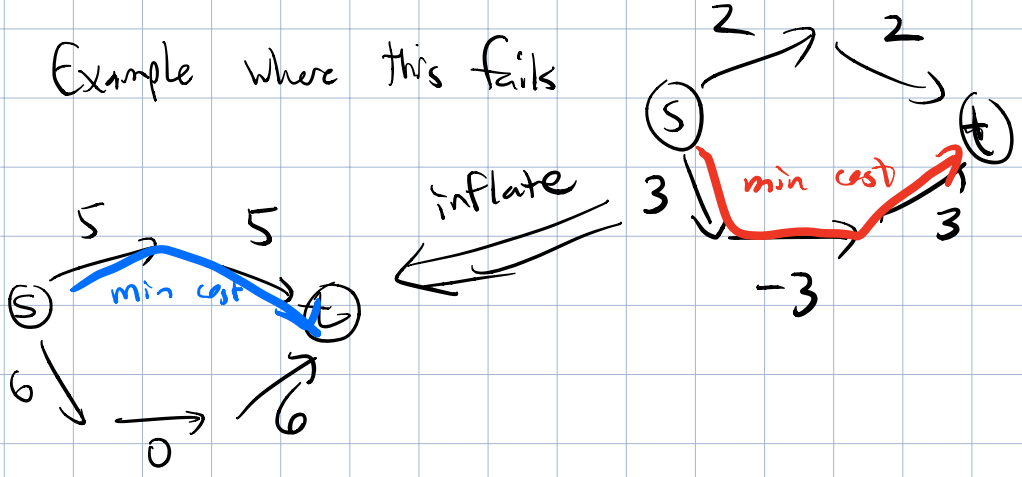
Dijkstra would start by picking (s,t) and would immediately terminate.

If Dijkstra instead backtracks wherever it discovers a cheaper path...



Can we just inflate the cost of every edge by a constant to make them positive, then solve using Dijkstra?

Example where this fails



A correct algorithm for shortest paths.

for $\ell = 0, 1, 2, \ldots, n-1$:
  for all $v \neq s$:
    compute the shortest walk from s to v consisting of $\leq \ell$ edges
  endfor
endfor
output the min-cost walk from s to t with $\leq n-1$ edges.

How?

How to implement the line in the red box?
Shortest s-v walk of $\leq \ell$ steps is either:
(a) shortest s-v walk of $\leq \ell-1$ steps, or
(b) shortest s-u walk of $\leq \ell-1$ steps followed by edge (u,v).

If MinCost$[v, \ell]$ stores minimum cost of a s-v walk with $\leq \ell$ steps and BestWalk$[v, \ell]$ stores a list containing the vertices in the min-cost walk, in order.

(*) justifies

$$(*)\quad MinCost[v,\ell] = \min\left\{ MinCost[v,\ell-1],\ \min_{(u,v)\in E}\left\{ Cost(u,v) + MinCost[u,\ell-1] \right\} \right\}$$

BellmanFord(G, cost, s, t):

let n = # of vertices.

initialize $MinCost[s,0]=0$, $MinCost[v,0]=\infty$ $\forall v \neq s$.

for $\ell = 1, \ldots, n-1$:

for all $v \in V$:

Compute $MinCost[v,\ell]$ using (*)

Compute $BestWalk[v,\ell]$ using the outcome of the (*) computation.

endfor

endfor

Discussion leading up to algorithm amounts to proof of correctness.

Running time:

$n-1$ iterations of outer loop

n iterations of inner loop.

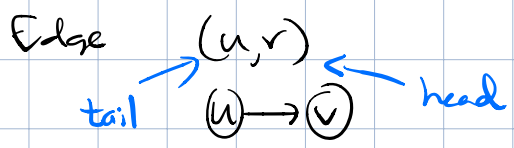
$O(n)$ work to evaluate (*)

$n \cdot (n-1)$ loop iterations each doing $O(n)$ work

$\Rightarrow$ $O(n^3)$ running time.

For sparse graphs with $m \ll n^2$ edges,

let $d^{in}(v)$ denote # edges whose head is v.



$$\sum_v d^{in}(v) = m.$$

To evaluate the formula (*) for vertex v takes $O(d^{in}(v))$ time.

Actual running time of B-F

$n-1$ iters of outer loop $\quad (n-1) \cdot O(m) = \boxed{O(mn)}$

n iters of inner loop.

Combined cost $O(\sum_v d^{in}(v)) = O(m)$