Dijkstra
Good: Fast! \( O(m \log n) \) run time
Bad: Need to maintain global heap (impossible for internet)
   - Fails with negative weights (see HW)

Bellman-Ford
- Dynamic Programming
- Slower but no global heap, negative weights OK

Bellman Ford:
**Input:** directed graph \( G = (V, E) \), edge weights \( l_e \), vertex \( s \in V \\
assume no negative cycles

**Output:** Shortest paths from \( s \) to all other \( v \in V \)
To Create D.P. (dynamic programming) algorithm:

1. Think of options for one (final) piece of optimal solution
2. Write optimal solution in terms of optimal solution to subproblem:

- **WMIS on line**: $V_n \in S$ or $V_n \notin S$

- **WMIS on tree**: in or out

- **LA or DC**

- **B**

- **A**
3. Create recurrence relation for objective function
4. Fill in array. **START** from base case! Go in opposite direction of recurrence
5. Work backwards to find optimal solution if desired.
Problem: on general graphs, hard to define subproblems

Idea: Edge present or not?

- Why this edge?
- Solution could change wildly if edge present
- Not simple recursive relationship

Q: What is shortest path from $s$ to $t$ with at most 2 edges? at most 3 edges?

A) 3, 1
B) 2, 0
C) 3, -1
D) 2, 1

We'll use max # of edges in path to order our subproblems
Q: If a graph $G$ has no negative weight cycles, how many edges are in the shortest path? (Let $|V| = n$, $|E| = m$)

A) no bound  
B) $m$  
C) $n$  
D) $n-1$

Proof: For contradiction, suppose the shortest path has more than $n-1$ edges.
- Then path must visit same vertex twice, so it contains a cycle.
- All cycles have non-negative weight, so if remove cycle from path, the result is a shorter path, a contradiction.
Let $P_{i,v} =$ shortest $s-v$ path with at most $i$ edges (or if no $s-v$ path) (assume unique $v$ if $i$)

**Case 1:** If $P_{i,v}$ has $\leq (i-1)$ edges,

$P_{i,v} = P_{i-1,v}$

**Case 2:** If $P_{i,v}$ has $i$ edges, then

$P_{i,v} = P_{i-1,v} + (w,v)$

for some $w : (w,v) \in E$

To prove, use proof by contradiction:

- **Case 1:** Suppose for contradiction $P_{i,v}$ has less than $i$ edges, but $P_{i,v} = P^* \neq P_{i-1,v}$. Then $P^*$ is a path with at most $i-1$ edges from $s$ to $v$. But $P_{i-1,v}$ is the shortest such path, so $\ell(P_{i-1,v}) \leq \ell(P^*)$, so you can always choose $P_{i,v} = P_{i-1,v}$ to get shortest possible path.
Case 2: Assume for contradiction $P_{i,v}$ has $i$ edges, but is $P_w + (w,v)$ for $P_w \neq P_{i-1,w}$.

... (rest is similar to Case 1)
Q: How many subproblems must be evaluated to calculate $P_{i,v}$?
A) $n+1$  B) $n$  C) $1 + |\{u: (u,v) \in E\}|$  D) $|\{u: (u,v) \in E\}|$

Case 1: $P_{i-1,v}$ (1 subproblem)

Case 2: $P_{i-1,w}$, for each $w \in \{u: (u,v) \in E\}$ (1 subproblem)

* Cycles are allowed – limit on $i$ keeps from infinite cycles

3 (Dynamic Programming) Create recurrence relation

Let $L_{i,v}$ be length of path $P_{i,v}$ ($\infty$ if no path)

Q: Base Case: $L_{0,s} = 0$  $L_{0,v} = \infty \ A \ v \in V-s$

Recurrence: $L_{i,v} = \min \left\{ \begin{array}{c}
L_{i-1,v} \\
\min_{(w,v) \in E} \left( L_{i-1,w} + l_{wv} \right) \end{array} \right\}$

Correctness: Using proof on previous page, $P_{i,v}$ must be related to one of $1 + |\{w: (w,v) \in E\}|$ subproblems. We look at all (exhaustive search)
Q: PseudoCode:

\[
L[i,s] = 0 \quad \forall i \\
L[0,v] = \infty \quad \forall v \in V-s
\]

for (i = 1 to n-1) \( \exists \) \( \text{max} \) \# edges required is n-1 \( \text{if no neg. cycles} \) (see HW)

for (v \in V-s)

\[
L[i,v] = \min \left\{ \min_{(w,v) \in E} \{ L[i-1,w] + l(w,v) \} \right\}
\]

\* Assume have inverse adjacency list \( A^{-1}_G[v] = \{ u : (u,v) \in E \} \)

\* Do for loop over \( A^{-1}_G[w] \)

Q: What is runtime of Bellman Ford? (Pick strongest bound)

A) \( O(n^2) \) \( \begin{array}{c} \text{B) } O(mn) \end{array} \) \( \begin{array}{c} \text{C) } O(n^3) \end{array} \) \( \text{D) } O(m^2) \)

\[
\sum_{i=1}^{n-1} \sum_{v} \left( 1 + \sum_{(w,v) \in E} 1 \right)
\]

\[
n(n+m) = O(nm)
\]

\( (n+m) \text{ if } G \text{ is connected} \)