10 November 2021

Dynamic Program Recap:

1. Formulate Problem Recursively
   (a) Specs: what is the input / desired output (this step has no how)
   (b) Recursive Solution
2. Build Solution
   (a) identify the subproblems
   (b) decide memoization data structure (at this point, doesn't need to be optimized)
   (c) determine dependencies
   (d) decide evaluation order
      → first, do the "simplest" subproblems; ie, those that do not depend on anything else
      → when problem X is in the eval order, all subproblems on which X depends must have already been solved.
   (e) RT / space requirements ← order?
   (f) Write it down!
   (g) improve time / space

All Pairs Shortest Path

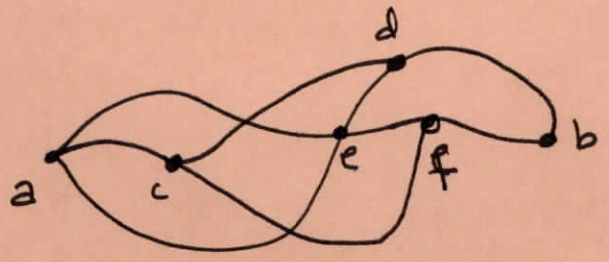
(1a) Input: $G = (V, E, w)$, a weighted graph

Output: (to recover shortest paths)
- `dist[n][n]`, an $n \times n$ array storing distances btwn nodes
  - undir setting: symmetric matrix
- `path`, an $n \times n$-dim array storing paths ~~from~~ btwn each pair of vertices
  - undir: `path[a][b] = rev(path[b][a])`
  - ↳ shortcut: `pred`, an $n \times n$ array that stores the 2nd to last vertex in the path

If you should be able to find `path[a][b]` from the pred 2d array.

(1b)

$$dist(a,b) = \begin{cases} 0, & a = b \\ \min\limits_{(x \to b) \in E} \{dist(u,x) + w(x \to b)\} \end{cases}$$



Problem:
- OK if in a DAG
- problematic b/c $a \to c \to f \to b \to d$ is considered as a subpath from $a \to b$

$$dist(a,b,\ell) = \begin{cases} 0, & a=b \text{ OR } \ell=0 \\ \infty, & \ell=0 \text{ and } a\neq b \\ \min\left\{ dist(a,b,\ell-1), \min_{(x\to b)\in E} \{dist(a,x)+\omega(x\to b)\} \right\} \end{cases}$$

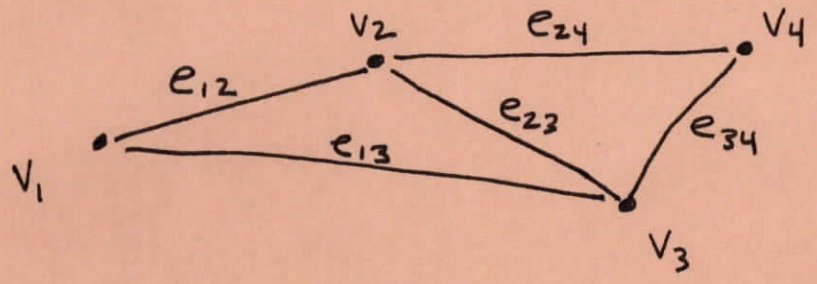
$dist(a,b,\ell)$: using at most ℓ edges; only need to consider up to ~~n~~ $n-1$.

Groups: ① Discuss 2a-2d

② Try it out on this example:

$\omega : E \longrightarrow \mathbb{R}$

$e_{12} \longmapsto 4.2$
$e_{13} \longmapsto 1.1$
$e_{23} \longmapsto 1.2$
$e_{24} \longmapsto 0.75$
$e_{34} \longmapsto 2.05$



(2a) $a \in V$, $b \in V$, $\ell \in$ ~~$\{0,$~~ $\{1,2,\dots,n-1\}$

(2b) "first pass data structure: 2D array"

$\ell=1$	V_1	V_2	V_3	V_4
V_1				
V_2				
V_3				
V_4				

$\ell=2$	1	2	3	4
1				
2				
3				
4				

$\ell=3$	1	2	3	4
1				
2				
3				
4				

$\ell=0$	1	2	3	4
1				
2				
3				
4				

Note: to compute $\ell=2$ "layer", just need to know $\ell=1$ layer