Recap: SSSP: Dijkstra (weighted, non-negative weights)

Runtime: $\Theta(|E|+|V|\log|V|)$ w/ Fibo. Heap

```
Dijkstra(G, w, s):
    For each v∈V:
        dist(v) = ∞
        parent(v) = ⊥
    dist(s) = 0
    For each v∈V:
        PQ.Push(v, dist(v)) // Add v w/ dist(v) as the key
    While !PQ.IsEmpty():
        u = PQ.ExtractMin()
        For v ∈ Adj[u]:
            If dist(v) > dist(u) + w(u,v):
                dist(v) = dist(u) + w(u,v)
                parent(v) = u
                PQ.DecreaseKey(v, dist(v))
```

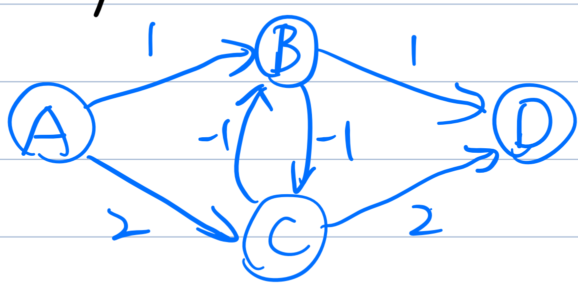
1. All Pairs Shortest Paths (APSP)

Instead of starting from a single source, want shortest paths between all pairs of vertices $u, v \in V$.

If weights are non-negative, we can run Dijkstra from each $u \in V$. Runtime is $\Theta(|V|^2 \log|V| + |V| \cdot |E|)$.

If weights are negative, we need to use Bellman-Ford instead, and the runtime becomes much worse $\Theta(|V|^2 \cdot |E|)$. We will improve this to $\Theta(|V|^3)$ using the Floyd-Warshall algorithm. (See textbook for Johnson's alg., achieving the better $\Theta(|V|^2 \log|V| + |V| \cdot |E|)$)

Remark: We allow negative weights, but assume no negative cycles.



← dist might be undefined!

$\Rightarrow$ Shortest paths are simple, and in particular have at most $|V|-1$ edges.

We assume input given as adj. matrix. So for $u, v \in V$,

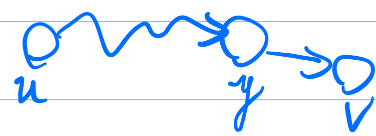
$$w(u,v) = \begin{cases} 0 & \text{if } u = v \\ \text{weight of } (u,v) & \text{if } u \neq v \text{ and } (u,v) \in E \\ \infty & \text{if } u \neq v \text{ and } (u,v) \notin E \end{cases}$$

Use DP!

<u>Attempt #1:</u>

Subproblems: For $u, v \in V$, the cost of shortest path from u to v using at most k edges.

Guess: next to last vertex



Recurrence:

$$DP(u,v,1) = w(u,v)$$

$$DP(u,v,k) = \min_{y \in V} \left(DP(u,y,k-1) + w(y,v) \right)$$

$DP(u, v, |V|-1)$ yields the final output for a given (u,v) pair.

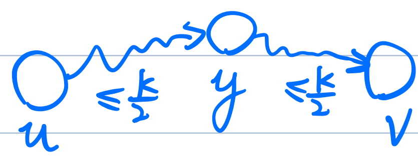
Runtime:

$\Theta(|V|^3)$ subproblems. $\Theta(|V|)$ time/subp.

$\Rightarrow \Theta(|V|^4)$ runtime

<u>Attempt #2:</u>

Guess middle vertex



Recurrence:

$$DP(u,v,1) = w(u,v)$$

$$DP(u,v,k) = \min_{y \in V} \left(DP(u,y,k/2) + DP(y,v,k/2) \right)$$

Runtime:

$\Theta(|V|^2 \log|V|)$ subp. $\cdot$ $\Theta(|V|)$ time/subp.

$\Rightarrow \Theta(|V|^3 \log|V|)$ runtime

Floyd-Warshall Algorithm:

Label the vertices $V = \{1, 2, \dots, |V|\}$.

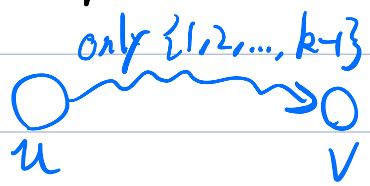
Subproblems: $\forall u, v \in V$, $k \in \{0, 1, 2, \dots, |V|\}$,
$DP(u, v, k)$ gives the min cost of paths from u to v that only use vertices from $\{1, 2, \dots, k\}$ as intermediate nodes.

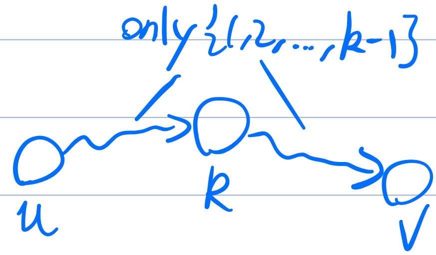
Recurrence:

$$DP(u, v, 0) = w(u, v)$$

$$DP(u, v, k) = \min\big(DP(u, v, k-1),\ DP(u, k, k-1) + DP(k, v, k-1)\big)$$

Only two possible cases:





Final output is $DP(u, v, |V|)$ for all u, v.

Runtime: $\Theta(|V|^3)$ subp. $\cdot$ $\Theta(1)$ time/subp.

$\Rightarrow \Theta(|V|^3)$ runtime!