All pairs shortest path (APSP)

Input: Directed graph $G = (V, E)$
weight fn $l : E \to \mathbb{R}$

Output: (1) shortest path between every pair of vertices,

OR

(2) Assert that the graph contains a cycle of negative weight.

Note: we have already seen that this is the best that we can do.

Obvious attempt:

(1) – For every choice of starting vertex run Bellman-Ford. $O(n^3)$

– What is the running time? $O(n^4)$ ✓

(2) Run Bellman-Ford from one source vertex. $O(n^3)$

Use 6.4.b from PS 3. $O(n \cdot (n+m) \log n) \le O(n^2)$

↳ What about negative weight cycles.

↳ Bellman-Ford already tells us?

Today: - an alternative algorithm that runs in time $O(mn)$.

- at most $O(n^3)$ since $m = O(n^2)$

Fun fact: - we do not know a significantly faster algorithm than this, e.g. something that runs in time $O(n^{2.999\ldots})$

- FW is from the 60's.

- Some people believe there is no such algorithm.

Fine grained complexity.

- a whole budding theory focussed around the existence / non-existence of a faster algorithm for APSP.

Floyd-Warshall's Algorithm

- Again based on Dynamic Programming.

So, what are the subproblems?
what is the optimal substructure?

_ clever choice
and hard to motivate _

Something to keep in mind ...

① – while setting up the DP, thinking about subproblems, substructure etc, might be helpful to think about the case that the graph does not have a negative weight cycle.

② – will later see, that the algorithms that comes out motivated by this case can be tuned a bit to detect negative cycles.

③ – for graphs with neg cycles, we don't care about computing shortest paths correctly.

① Let the vertices in the graph be numbered as $\{1, 2, 3, \ldots, n\}$ ∴ Denote this by set V.

② For every choice of vertices v, w and 'index' $k \in \{1, 2, \ldots, n\}$

$L_{k,v,w}$:= length of the shortest path in G that

① starts at v

② ends at w

③ uses only vertices in the subset $\{1, 2, 3, \ldots, k\}$ internally

(4) does not contain a directed cycle.

- Contrast with the DP in Bellman-Ford.

If no such path, set $L_{k,v,w} = \infty$

How many subproblems?

- What is the optimal substructure?

Optimal substructure

Let P : v-w path with no cycles, all internal vertices in $\{1, 2, 3, \dots, k\}$
- shortest such v-w path.

So, $L_{k,v,w} = \sum_{e \in P} l_e$.

What does P look like?

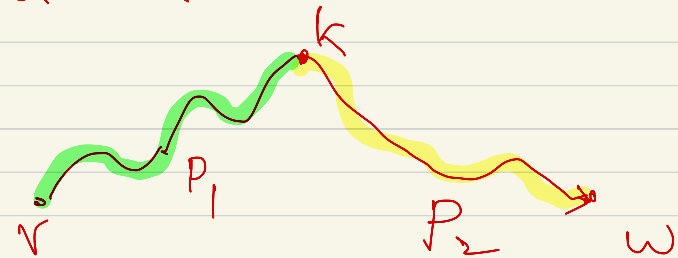
Let u = internal vertex with the largest index in P.

Know, $u \le k$.

<u>Option 1</u>: $u < k$.

$\Rightarrow P$ is a soln to the smaller subproblem: $L_{k-1,v,w} = \sum_{e \in P} l_e$.

Option 2: $u = k$



- what can we say about internal vertices in green and yellow subpaths?

- All these internal vertices are in $\{1, 2, \ldots, k-1\}$.

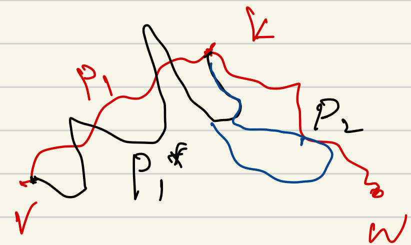
what about optimality of P_1, P_2 as paths between $v \to k$ and $k \to w$ with internal vertices in $\{1, 2, 3, \ldots, k-1\}$?

Claim:

- they are optimal if G does not have any negative weight cycles.

Pf: P_1 is an optimal s-k path with internal vertices in $\{1, 2, \ldots, k-1\}$, cycle free

- Suppose not.
- Replace P_1 by the optimal such path - P_1^*



$$\text{len}(\underbrace{P_1^* \circ P_2}) < \text{len}(P_1 \circ P_2) = \text{len}(P)$$

- ↳ may not be cycle free
- cut off the cycle to get a cycle free path.
- the overall length does not increase] only true if no neg weight cycles.

□

- Again, recall that for graphs with negative cycles, we don't care about shortest path calculations, just detecting that there is a neg. cycle.

- So, opt. substructure etc does not really matter.

To summarize.

Lemma: Let G have no negative weight cycles.

Let P be a shortest cycle free $v \to w$ path in G, with all the internal vertices in $\{1, 2, \ldots, k\}$.

Then either

① P has all its internal vertices in $\{1, 2, \ldots, k-1\}$

OR

② P is a concatenation of P_1, P_2 where

P_1 = shortest cycle free $v \to k$ path with internal vertices in $\{1, 2, \ldots, k-1\}$

P_2 - shortest cycle free $k \to w$ path with internal vertices in $\{1, 2, \ldots, k-1\}$.

Proof:

Corollary:

Recurrence:

$$L_{k,v,w} = \min \begin{cases} L_{k-1,v,w} \\ L_{k-1,v,k} + L_{k-1,k,w} \end{cases}$$

$\boxed{\checkmark}$

What are the base cases? $k = 0, 1, 2, \ldots$

Floyd-Warshall Algorithm

$A := (n+1) \times n \times n$ matrix

* v could equal w in these loops

1. for $v \in V, w \in V$,

   if $v = w$, $A(0, v, w) := 0$

   else, if $(v, w) \in E$, $A(0, v, w) := \ell_{vw}$

   else, $A(0, v, w) := \infty$

2. for $k = 1$ to n

   for $v \in V, w \in V$

$$A(k, v, w) := \min \begin{cases} A(k-1, v, w), \\ A(k-1, v, k) + A(k-1, k, w). \end{cases}$$

- So far, as discussed the above pseudocode should correctly solve the problem in negative cycle free graphs.
- Now: identifying graphs with negative cycles.

③ for $v \in V$,
    if $A(n, v, v) < 0$
        Return "Negative cycle";

Return A.

Correctness:

If no negative cycle then Blocks 1 and 2 correctly compute the shortest distance between every pair. And, $A(n,v,v) \geq 0$ for all $v \in V$.

All that remains is:- - -

Lemma: If G has a negative weight cycle, then, there is a vertex $v \in V$ s.t at the end of blocks 1 and 2 in F-W algorithm we have $A(n,v,v) < 0$.

Proof:

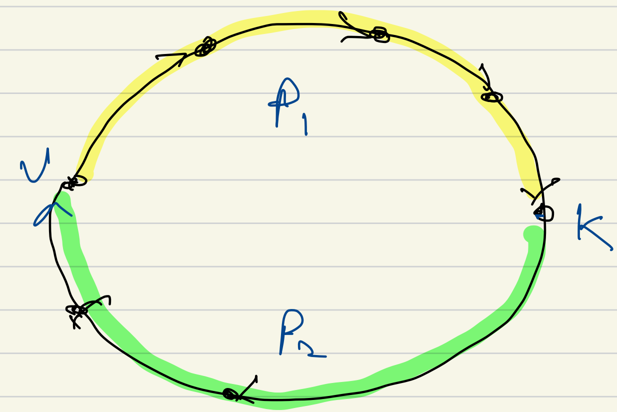
Block 1: initialization.
makes sense even for ghs with neg. cycles (and no self loops)

What does Block 2 really compute in graphs with negative weight cycles?
Recall: proof of Lemma / Recurrence relied on no neg. cycles.

- Suppose that G has a negative weight cycle.
- Consider one such cycle with no repeating vertices.
- Why is there always such a neg wt cycle?

Let k = vertex with largest index in C

v = any vertex apart from k in C



claim: For a graph with neg cycles,
$\forall v, w \in V, k \in \{0, 1, \ldots, n\}$,

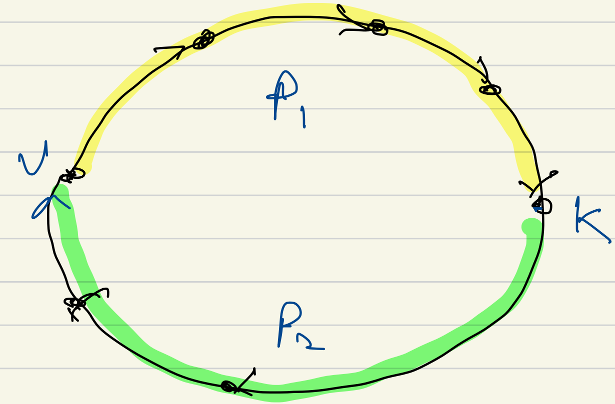
$A(k, v, w) \leq$ Length of shortest $v \to w$ path in G, that is cycle free and has internal vertices in $\{0, 1, \ldots, k\}$. $(L_{k,v,w})$

- Note the inequality here.
- An equality if G has no negative weight cycles.

Will see a proof of the claim later. First

claim $\Rightarrow$ Lemma.

Note: P_1, P_2 are cycle free paths with internal vertices in $\{1, 2, \ldots, k-1\}$



So, from claim,

$$A(k-1, v, k) \leq \sum_{e \in P_1} l_e$$

$$A(k-1, k, v) \leq \sum_{e \in P_2} l_e$$

$$\Rightarrow A(k-1, v, k) + A(k-1, k, v) \leq \sum_{e \in C} l_e < 0$$

From Block ② of FW,

$$A(k, v, v) \leq A(k-1, v, k) + A(k-1, k, v) < 0.$$

And, $A(n, v, v)$ is equal or smaller than

$A(k, v, v) \quad \forall k \leq d.$

$\Rightarrow A(n, v, v) < 0.$ $\boxed{}$

Now, proof of claim:

— induction on k.

Base case ....

Induction step: Assume correct for $i \in \{0, 1, \ldots, k-1\}$

Prove it for $i = k$.

In the F-W Algorithm.

$$A_{k,v,w} := \min \begin{cases} A_{k-1,v,w} \\ A_{k-1,v,k} + A_{k-1,k,w}. \end{cases}$$

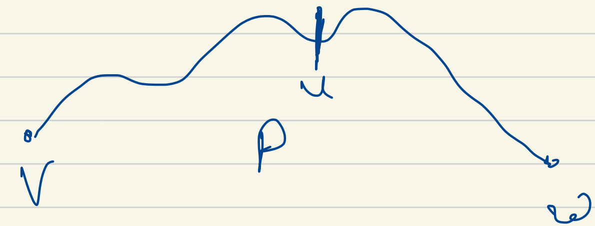
Will argue: (1) $A_{k-1,v,w} \le L_{k,v,w}$.

(2) $A_{k-1,v,k} + A_{k-1,k,w} \le L_{k,v,w}$.

Together (1) and (2) imply,
$A_{k,v,w} \le L_{k,v,w}$.

(1) follows from induction hypo. $A_{k-1,v,w} \le L_{k-1,v,w}$
and $L_{k-1,v,w} \le L_{k,v,w}$
(Defn. of L)

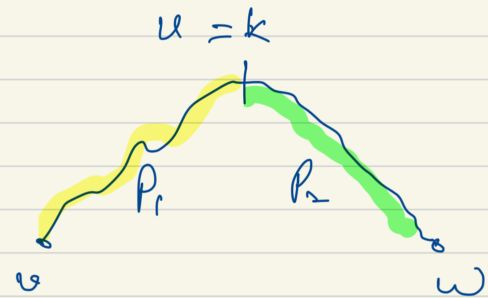
$P :=$ shortest cycle free v–w path in G with internal vertices in $\{1, \ldots, k\}$.



$u :=$ internal vertex with largest index.

<u>Case 1</u> <u>$u < k$.</u>

Case 2 $u = k$.



Earlier we argued that P_1, P_2 must be opt cycle free paths between $v \to k$ and $k \to w$ resp. with internal vertices in $\{0, 1, \ldots, k-1\}$.

Is that still true — when G has a neg weight cycle?

Now,

$$L_{k,v,w} = \sum_{e \in P} l_e$$

$$= \sum_{e \in P_1} l_e + \sum_{e \in P_2} l_e$$

$$\geq L_{k-1,v,k} + L_{k-1,k,w}.$$

$$\geq A(k-1,v,k) + A(k-1,k,w). \qquad \text{— (2)}$$

In F-W.

$$A_{k,v,w} = \min \begin{cases} A_{k-1,v,w} \\ A_{k-1,v,k} + A_{k-1,k,w} \end{cases}$$

From (1) and (2), both these quantities are

$\leq L_{K,V,W}$. $\boxed{}$

Q. Knapsack with negative weights and prices.