Thursday, April 1, 2021 10:05 AM

Dynamic Programming

↳ powerful & general purpose

1) break down the full task into smaller subproblems

slightly smaller vs. (1/2) small in divide & conquer

Divide & Conquer

significant reduction

2) there is lot of overlap between the subproblems

⇒ subproblems can be solved once & reused in larger subproblems

no overlap between subproblems

3) Used for optimization problems (e.g., shortest path, min length path)

↳ each subproblem should able to be be solved optimally

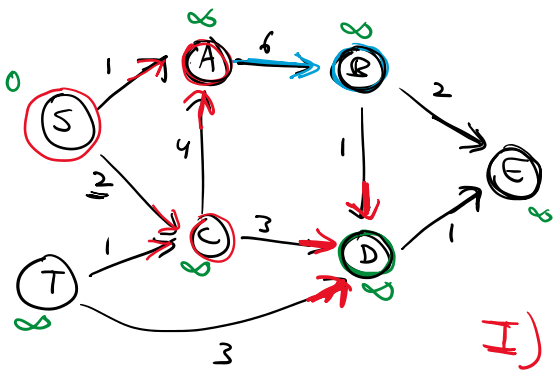
Implicity there is a DAG over the subproblems

Shortest paths in DAGs

Alg 1: topological sort & trace a path from the given starting node to other nodes

$O(|V| + |E|)$

Alg 2: Dynamic programming formulation



given starting vertex S

I) $d(u) \overset{\text{def}}{=}$ shortest distance from s to u

Task: Compute $d(u)$ $\forall u$

$d(A) = \min \{ d(s) + w(s,A),\ d(C) + w(C,A) \}$

$d(A) = \min \begin{cases} d(s) + w(s,A) \\ d(c) + w(c,A) \end{cases}$

$d(B) = d(A) + w(A,B)$

look at $(A \to B)$

$d(D) = \min_{(v,D)} \begin{cases} d(B) + w(B,D) \\ d(C) + w(C,D) \\ d(T) + w(T,D) \end{cases}$

here $v = \{B, C, T\}$

... from s to u

Task: Compute $d(u)$ $\forall$ u

II) base case: initialization

$$d(u) = \begin{cases} 0 & \text{if } u = s \\ \infty & \text{if } u \neq s \end{cases}$$

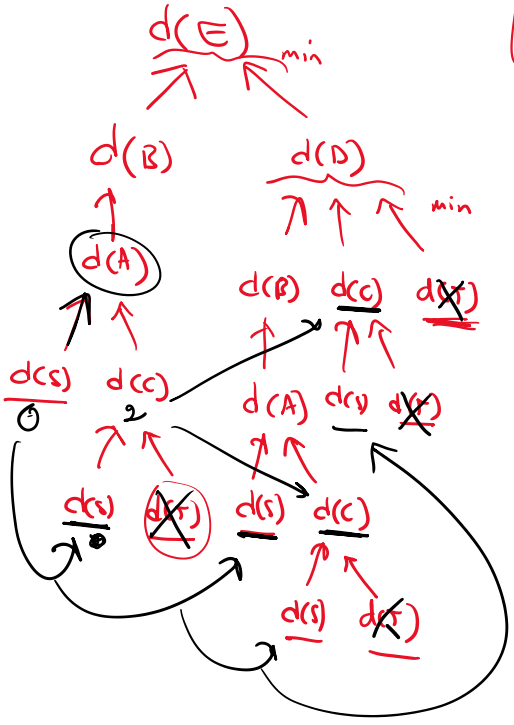
III) subproblems: recursive equation (written "backwards")

$$d(u) = \min_{(v,u) \in E} \{ d(v) + w(v,u) \}$$

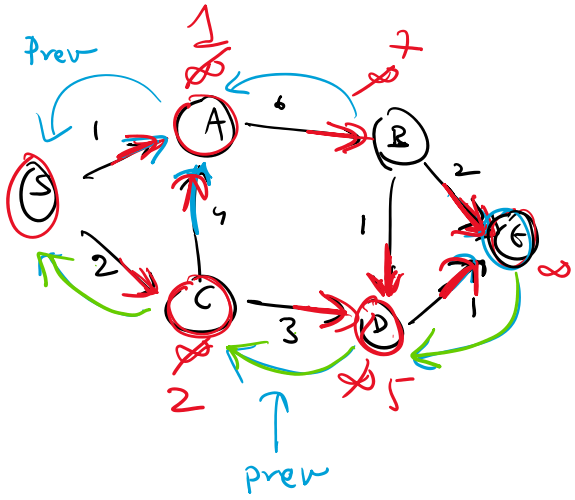
u's — minimum over incoming edges — subproblem

IV) Solve in forward manner i.e. start from the base case & solve bigger subproblems

recursion tree



d(T) discard since S is start



$d(s) = 0$

$(1, s) = d(A) = \begin{cases} d(s) + w(s,A) = 0+1 \\ d(c) + w(c,A) = 2+4 = 6 \end{cases}$

$(?, A) = d(B) = d(A) + w(A,B)$
$= 1 + 6 = 7$

$(6, D) = d(E) \begin{cases} d(B) + w(B,E) = 7+2 = 9 \\ d(D) + w(D,E) = 5+1 = 6 \end{cases}$

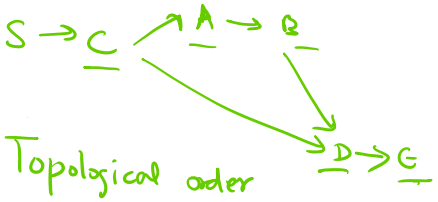
$(S, C) = d(D) = \{ d(C) + w(C,D) = 2+3 = 5$

$d(E) = 6$

path = D, C, S

d(E) = 6

path = D, C, S



Topological order

Implicit order followed by DP (Dynamic Programming)

$(S, C) = d(D) = \begin{cases} d(C) + w(C,D) = 2+3 = 5 \\ d(B) + w(B,D) = 7+1 = 8 \end{cases}$

min Value

arg min

$d(C) = d(S) + w(S, C)$

$(2, S)$

$= 0 + 2 = 2$

Complexity?

$O(|V|)$ $\quad d(u) = \begin{cases} 0 & u = s \\ \infty & u \neq s \end{cases} \quad \forall u$

$O(|E|)$ time

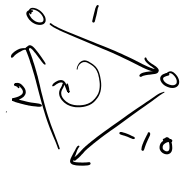
$$d(u) = \min_{(v,u) \in E} \{ d(v) + w(v,u) \} \quad \forall u$$

we need to compute $O(|V|)$ terms, i.e. $d(u) \ \forall u \in V$

we looked @ DAGs $\rightarrow O(|V| + |E|)$ time

All pairs shortest paths

negative cycle



Positive cycle

G: weighted directed graph

allow negative weights

(do not allow negative cycles)

$d(u,v) \ \forall u, \forall v \in V$

APSP is to find the shortest path between all pairs of vertice

Bellman Ford (G, s)

$d(s,u) = \infty \ \forall u, \quad d(s,s) = 0$

for $t = 0, 1, \ldots, |V|$

$O(|V| \cdot |E|)$ { for all $e = (u,v) \in E$

$O(|V| \cdot |E|)$ { for all $e = (u,v) \in E$ { $d(s,v) = \min \{ d(s,u), d(s,v) + w(v,u) \}$ }

Alg1: APSP_BellmanFord(G):

$O(|V|) \rightarrow \forall s \in V$ → call BellmanFord(G, s) $O(|V| \cdot |E|)$] $d(s,u)$ $\forall s \in V$ $\forall u \in V$

total time: $O(|V|^2 \cdot |E|)$ → Sparse G: $|E| = O(|V|)$ $O(|V|^3)$ ✓ (best case)

→ dense G: $|E| = O(|V|^2)$ (worse case) $O(|V|^4)$ ✗

Alg2: allow at most m edges for any path

$m = 1, 2, \ldots, |V|-1$

I) $\ell^m(i,j) \equiv \ell^m_{ij} \equiv$ length of the shortest path from i to j that uses at most m edge

We need to compute ℓ^m_{ij} $\forall i \in V$, $\forall j \in V$ } i, j are vertices

$\forall m = 0, 1, \ldots, |V|-1$ ← max edge on path

II) base: Initialization

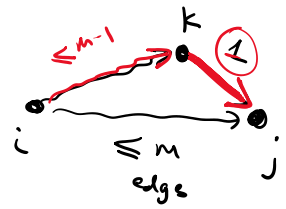
$$\ell^1_{ij} = \begin{cases} w(i,j) & \text{if } (i,j) \in E \\ \infty & \text{if } (i,j) \notin E \end{cases}$$

III): recursive equation

ℓ^m $\left(n^m \right)$

III). recursive equation

$$l_{ij}^{m} = \min_{(k,j)\in E} \left\{ l_{ij}^{m} \;,\; l_{ik}^{m-1} + w(k,j) \right\}$$

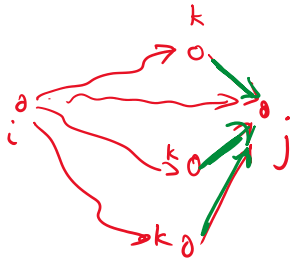


$$prev_{ij}^{m} = \arg\min \{ \quad // \quad \}$$

IV) Extract the all pairs shortest paths from $l_{ij}^{|V|-1}$

Initialize

$$l_{ij}^{1} = \begin{cases} w(i,j) & \text{if } (i,j) \in E \\ \infty & \text{otherwise} \end{cases}$$



$O(|V|^3)$

```
for m = 2, 3, ..., |V|-1
    for i = 0 ... |V|-1
        for j = 0 ... |V|-1
            l_ij^m = min_(k,j) { l_ij^m , l_ik^(m-1) + w(k,j) }
            prev_ij^m = arg min_(k,j) { // }
```

$O(|V|)$

$O(|V|^4)$ worse case (dense graphs) ✗

$O(|V|^3)$ beer case (sparse graphs) ✓

Alg 3: Instead of restricting the max # of edge to m

we will restrict the set of allowd vertice

$A = \phi$ ← initially

$A = \{0\}$ ← set of allowed vertice on a shortest path

$A = \{0\}$ ← set of allowed vertices on a shortest path

$A = \{0, 1\}$

$\vdots$

$A = \{0, 1, \ldots, |V|-1\}$

$|V| = n$

vertices $= \{v_0, v_1, \ldots, v_{n-1}\}$

$= \{0, 1, \ldots, n-1\}$

1) define the objective function

$\ell_{ij}^{k} = d(i, j, k) \equiv$ shortest path from i to j using the allowed set $A = \{0, 1, \ldots, k\}$

$A = \{v_0, v_1, \ldots, v_k\}$

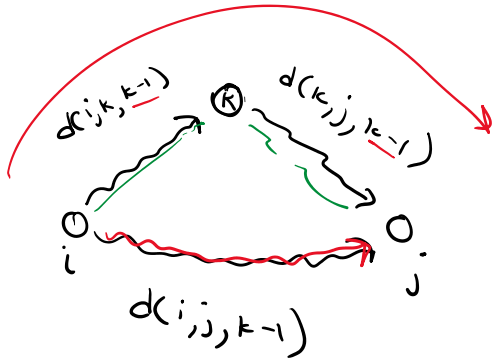
2) base case $-1 \equiv$ empty set $= A = \phi$

$d(i, j, -1) \equiv$ direct edge $= \begin{cases} w(i,j) & \text{if } (i,j) \in E \\ \infty & \text{if } (i,j) \notin E \end{cases}$

3) recursive equation in terms of subproblems

before: $A' = \{0, 1, \ldots, k-1\}$

after $A = \{0, 1, \ldots, k\}$



$$d(i,j,k) = \min \begin{cases} d(i,j,k-1), \\ d(i,k,k-1) + d(k,j,k-1) \end{cases}$$

$d(k,j,k-1)\big)$

final answers: $d(i,j,|V|-1)$

Initialize

$$d(i,j,-1) = \begin{cases} w(i,j) & \text{if } (i,j) \in E \\ \infty & \text{otherwise} \end{cases}$$

$O(|V|^3)$

```
for k = 0, 1, ..., |V|-1
    for i ∈ V
        for j ∈ V
            d(i,j,k) = min { d(i,j,k-1),
                             d(i,k,k-1) + d(k,j,k-1) }
```

2 options → $O(1)$

total time $O(|V|^3)$ ← worse case time!

Floyd Warshall Algorithm

Exam II

- Chapter 2: Only sec 2.6 poly multiplication
- Chapter 3: DFS
- Chapter 4: BFS
- Chapter 5: Only sec 5.1 and 5.2
  - 5.1 ↑ MST
  - 5.2 ↑ Huffman