CS 252

F, 17 May 2024

Q — writing a runtime recurrence for a recursive alg.

```
def sum(n, a):
    if n == 0:
        return 0
    else
        return a[n] + sum(n-1, a)
```

$T(n)$ = runtime

$$\begin{cases} T(n) = T(n-1) + c \\ T(0) = 0 \end{cases}$$

Same Q, D + C

~~n = len(a)~~

n = right - left

```
def BigSubarraySum(left, right, a)
    sum1 = BSS(left, middle, a)
    sum2 = BSS(middle+1, right, a)
    sum3 = linear time WIF
    return max(sum1, sum2, sum3)
```

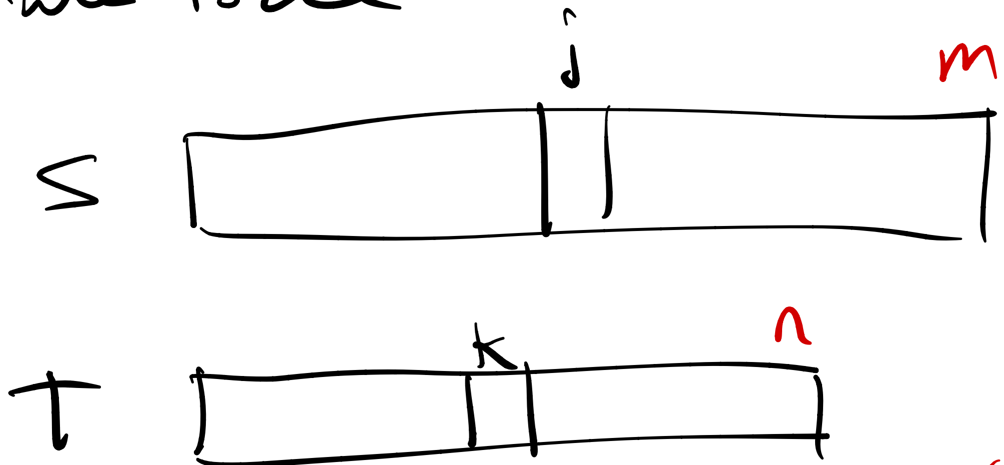
$T(n)$ runtime

$T(1) = c$

$T(n) = 2T(n/2) + dn$

Good runtime for #3?

Brute force



$O(m \cdot n)$

for each (j,k)

What's the longest substring ending at $S[j]$ & $T[k]$

scan backwards $O(m+n)$

Overall $O((m+n)mn)$

Given: directed weighted graph G
- weights can be negative
- no negative cycles
- weight on edge $(u,v) \in E$ is denoted c_{uv}
  "cost" u to v

Goal: find lowest-weight path from any given source node $\underline{s}$ to target node $\underline{t}$

Strategy: dynamic programming

Given target node t

Define

$$M[k, u] \quad \begin{pmatrix} 0 \le k < n \\ u \in V \end{pmatrix}$$

= the lowest cost of a path from u to t w/ $\le k$ edges

$M[k, u]$ = lowest cost u-to-t, $\leq k$ edges

$M[0, u] = \infty$ for $u \neq t$

$M[k, t] = 0$ for all k

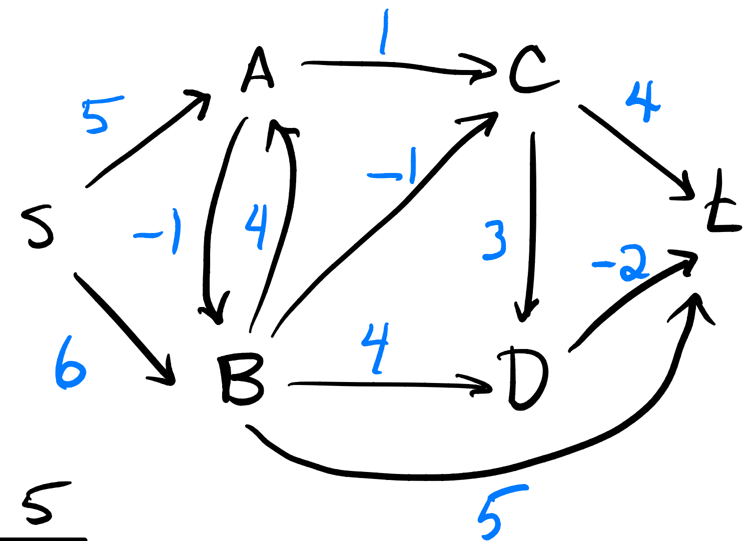
$$M[k, u] = \min \begin{cases} M[k-1, u] \\ \min\limits_{(u,v) \in E} M[k-1, v] + c_{uv} \end{cases}$$

$M[k, u]$ = lowest cost u-to-t, $\leq k$ edges

$M[0, u] = \infty$ for $u \neq t$

$M[k, t] = 0$ for all k

$$M[k,u] = \min \begin{cases} M[k-1, u] \\ \min\limits_{(u,v) \in E} M[k-1, v] + c_{uv} \end{cases}$$



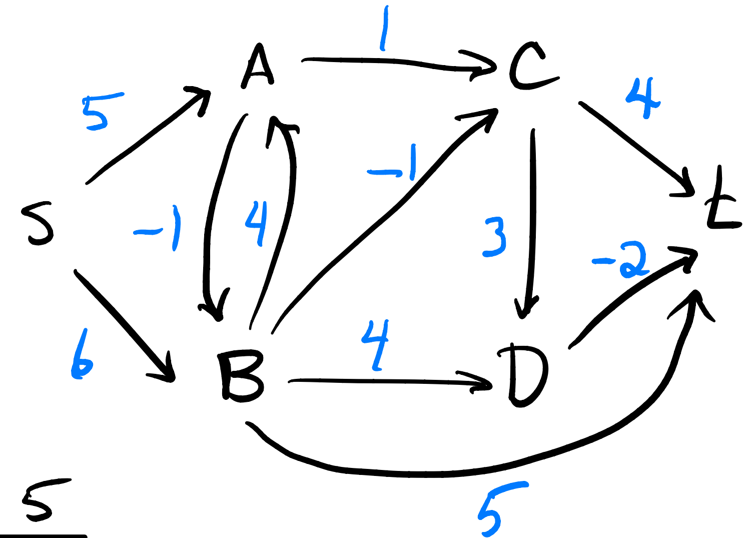
	0	1	2	3	4	5
t	0	0	0	0	0	0
A	∞	∞	4			
B	∞	5	2			
C	∞	4	1			
D	∞	-2	-2			
S	∞	∞	11			

$M[k,u]$ = lowest cost u-to-t, $\leq k$ edges

$M[0,u] = \infty$ for $u \neq t$

$M[k,t] = 0$ for all k

$$M[k,u] = \min \begin{cases} M[k-1,u] \\ \min\limits_{(u,v)\in E} M[k-1,v] + c_{uv} \end{cases}$$



	0	1	2	3	4	5
t	0	0	0	0	0	0
A	∞	∞	4	1	-1	-1
B	∞	5	2	0	0	0
C	∞	4	1	1	1	1
D	∞	-2	-2	-2	-2	-2
S	∞	∞	11	8	6	4

$O(|V||E|)$

$|V| = n$, $|E| \leq n\cdot(n-1)$

Dijkstra w/ minheap

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

B-F

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

DP recap

- What are we trying to optimize or count?
- What does our search space look like?
-

Make change

Given coins = [1, 5, 10, 25]

Given amount D

Smallest # coins ?

$$M[n] = \min_{c \in \text{coins}} 1 + M[n-c]$$

Minimum edit distance recap

		s	p	o	k	e
	0	1	2	3	4	5
b	1	1	2	3	4	5
o	2	2	2	2	3	4
o	3	3				
k	4					

min cost to transform "spo" to "b"

"spo" to "bo"

Minimum edit distance recap

		s	p	o	k	e
	0	1	2	3	4	5
b	1	1	2	3	4	5
o	2	2	2	2	3	4
o	3	3				
k	4					