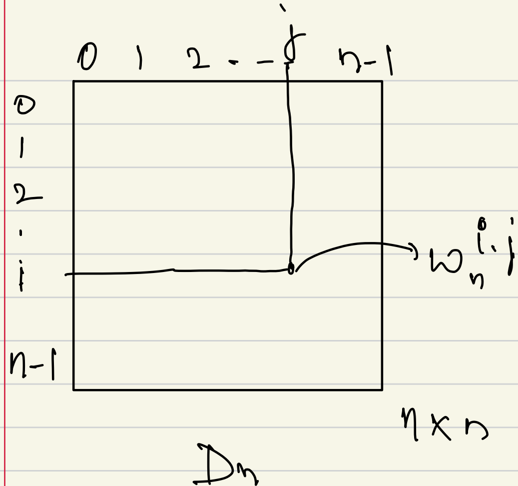
Feb 11, 2022

Lecture 11

Agenda:

- Discussion of problem set 3
- Most questions will be discussed by Kushagra and Roshan
- Q 2 is discussed here.

Q.2 Factoring the DFT matrix as a product of sparse matrices.



For a vector $v \in \mathbb{C}^n$, $v = (v_0, v_1, \ldots, v_{n-1})$

$$\boxed{D_n \cdot v} = \left(f(\omega_n^0), f(\omega_n), f(\omega_n^2), \ldots, f(\omega_n^{n-1}) \right)$$

where $f(x) = v_0 + v_1 x + v_2 x^2 + \cdots + v_{n-1} x^{n-1}$

Divide and conquer algorithm for computing $D_n \cdot v$

$$f_{even}(x) := v_0 + v_2 x + v_4 x^2 + \cdots$$

$$f_{odd}(x) := v_1 + v_3 x + v_5 x^2 + \cdots$$

$\forall x$

$$f(x) = \underline{f_{even}(x^2)} + x \cdot \underline{f_{odd}(x^2)}$$

So, for every i

$$\begin{bmatrix} f(w_n^i) \\ \\ \\ \end{bmatrix} = f_{even}(w_n^{2i}) + w_n^i \cdot f_{odd}(w_n^{2i})$$

$$\deg(f_{even}), \deg(f_{odd}) = n/2$$

$$\underline{D_n} \cdot \underline{v} = \begin{bmatrix} & & \\ & & \end{bmatrix} \underbrace{\begin{bmatrix} \\ \\ \end{bmatrix}}_{v}$$

✓ ① Go from f to f_{even}, f_{odd} ✓

✓ ② Evaluate f_{even}, f_{odd} at all $n/2$ th roots of unity

✓ ③ Combine the info.

Express ①, ②, ③ as linear transforms M_1, M_2, M_3
(matrices)

$\underline{\forall v}$

$$\underline{D_n} \cdot v = (M_3 \cdot M_2 \cdot M_1) \cdot v$$

$$\Rightarrow D_n = M_3 \cdot M_2 \cdot M_1$$

Step 1 : $f \rightarrow f_{even} \,,\, f_{odd}$

$$\underset{f_{even}}{\begin{bmatrix} \\ \\ \\ \\ \end{bmatrix}} = \underset{A_{even}}{\begin{bmatrix} 1 & 0 & 0 & - & - & 0 \\ 0 & 0 & 1 & 0 & - & - \\ 0 & 0 & 0 & 0 & 1 & - \\ & & \vdots & & & \end{bmatrix}}_{n/2 \times n} \begin{bmatrix} v_0 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{bmatrix}$$

$$f_{odd} = \underset{n/2 \times n}{A_{odd}} \cdot v$$

$$M_1 = \begin{bmatrix} A_{even} \\ \hline A_{odd} \end{bmatrix}$$

$$M_1 \cdot V = \begin{bmatrix} f_{even} \\ f_{odd} \end{bmatrix}$$

← Exactly 1 non-zero in every row of M_1

Step 2 Eval f_{even}, f_{odd} at $n/2$ th roots of 1.

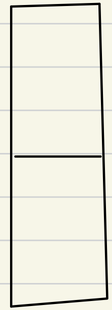
$$\underbrace{\begin{bmatrix} \quad & \quad \\ \quad & \quad \end{bmatrix}}_{D_{n/2}\ \ \frac{n}{2}\times\frac{n}{2}} \underbrace{\begin{bmatrix} \quad \\ \quad \end{bmatrix}}_{f_{even}}$$

$$M_2 = \underbrace{\begin{bmatrix} D_{n/2} & 0 \\ 0 & D_{n/2} \end{bmatrix}}_{n \times n} \begin{bmatrix} f_{even} \\ f_{odd} \end{bmatrix}$$

→ Not row sparse

Step 3.

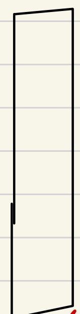
Input.



→ eval of f_{even} at $n/2$ roots of 1

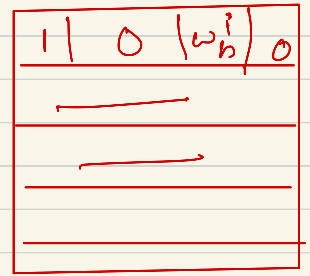
→ eval of f_{odd} at $n/2$ roots of 1

Output



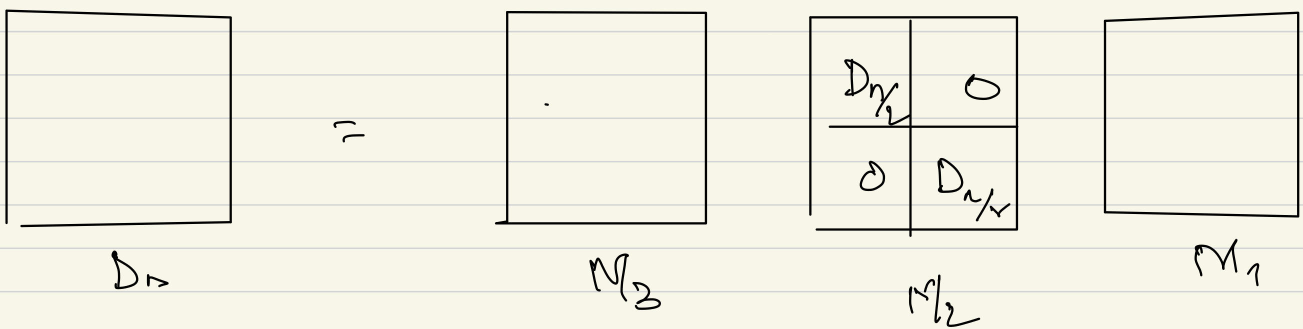
→ eval of f at nth roots

$$f(w_n^i) = f_{even}(w_n^{2i}) + w_n^i \cdot f_{odd}(w_n^{2i})$$



→ every row has 2 non-zero entries

Overall.



2-non-zero entries

1-non-zero entry

- Recurse on $D_{n/2}$ for $O(\log n)$ steps.

Q 4.

(b) Negative edge weights — so can't use Dijkstra's.

Strategy

① Make the edge weights non-negative while preserving the shortest path

② Will use the fact that we are given min dist from $v \to t$ $\forall v \in V$.

Part (a)

$\forall (u,v) \in E$

$$\begin{cases} d(u) \leq l_{(u,v)} + d(v) \\ \Rightarrow \; l_{(u,v)} + d(v) - d(u) \geq 0 \end{cases} \quad \forall (u,v) \in E.$$

$\forall$ edges $(u,v) \in E$

set $\boxed{\tilde{\ell}_{(u,v)} := \ell_{(u,v)} + d(v) - d(u)}$

Crucially: $\tilde{\ell}_{(u,v)} \geq 0 \quad \forall (u,v) \in E$.

(2) Run Dijkstra using $\tilde{\ell}$.

Claim: $\forall u \in V$, shortest (u,t') path in $(G, \tilde{\ell})$ is the same as the shortest (u,t') path in (G, ℓ).

(weights might change)

Pf: Path in G

$$u \to v_1 \to v_2 \to \cdots \to v_k \to t'$$

$$\mathrm{Len}(P) = l_{(u,v_1)} + l_{(v_1,v_2)} + l_{(v_2,v_3)} + \dots + l_{(v_{k-1},v_k)} + l_{(v_k,t')}$$

$$\widetilde{\mathrm{Len}}(P) = \tilde{l}_{(u,v_1)} + \tilde{l}_{(v_1,v_2)} + \widetilde{} \qquad \widetilde{} \qquad + \tilde{l}_{(v_k,t')}$$

$$= \left(\underline{l_{(u,v_1)}} + d(\cancel{v_1}) - d(u) \right)$$

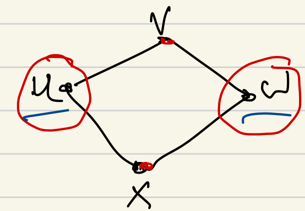
$$+ \left(\underline{l_{(v_1,v_2)}} + \cancel{d(v_2)} - \cancel{d(v_1)} \right)$$

$$= \mathrm{Len}(P) + \boxed{d(t') - d(u)}$$

Q.6. Detecting a 4 cycle.

Algo 1:

n^2 ① Iterate over all pairs $(u,w) \in V \times V$, $u \neq w$



n ② check if they share at least 2 common neighbors.

↦ Iterate over all vertices $v \in V$, $v \neq u, w$

↦ Check if $(u,v), (v,w) \in E$

↳ Increase a counter.

$O(|V|^3)$ time.

Algo 2

A = adjacency matrix of G, $n \times n$ matrix

A^2 —

$(A^2)^2 = A^4$

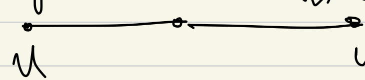
2 $n \times n$ matrix mult

$O(n^3)$ time trivially

(Faster using Strassen's algorithm)

$$(A^2)_{(u,v)} = \sum_{w \in V} A_{(u,w)} \cdot A_{(w,v)}$$

= # walks of length 2 from $u \to v$ in G

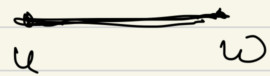


Walk vs Path

can repeat vertices/edges

"walk with no repetitions"

$(A^2)_{(u,u)}$

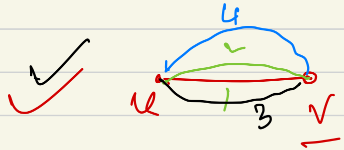


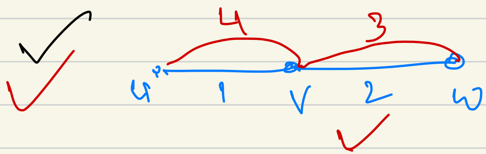
By Induction on i

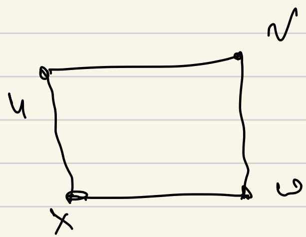
$(A^i)_{(u,v)}$ = # walks from u to v in graph G of length i.

$(A^4)_{(u,v)}$ = # walks of length 4 from u, v in G.

$(A^4)_{(u,u)}$ = # walks of len. 4 from u — u







4-cycles.

Plan:

If we can subtract the contribution of 4-len walk that are not 4-cycles, we would be done.

Q.3: Steiner Tree

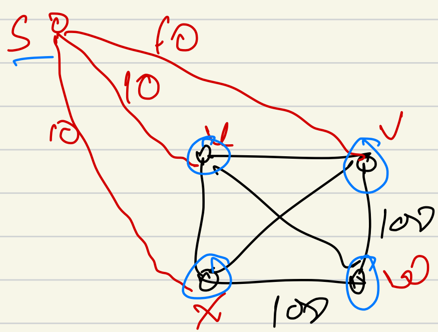
$G(V, E)$, positive edge costs.

$X \subseteq V \rightarrow$ set of terminals.

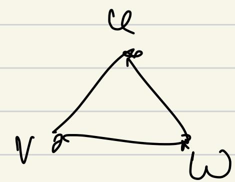
- Want the min weight subgraph of G that contains $\underline{X}$.
  ↳ connected

- A tree without loss of generality

Could contain other vertices.



Also given: edge weights satisfy triangle inequality



$$w(u,v) + w(u,w) \geq w(v,w)$$

— 'Metric' Steiner Tree.

Wanted: an algo for metric steiner tree in time

$$(f(k) \cdot n^c)$$

where c = constant (indep of every other params)
f = arbitrary function
k = # terminals

Known: DP based algo for General Steiner tree

- $f(k) = 3^k$
- $c = 2$

$\Big\}$ $O(3^{\#\text{terminals}} \cdot n^2)$

- Dreyfus-Wagner.