Lecture 4:

Sublinear time algorithms for coloring graphs

Graph Coloring

<u>def</u> a proper c-coloring of G
assigns color c_v from "palette" $\{1..c\}$
to each $v \in V$ s.t. $\forall (u,v) \in E$ $c_u \neq c_v$

- in general, NP-complete
- important case solvable in linear time "$\Delta+1$ - coloring"

max degree Δ
$c = \Delta+1$

runtime: $O(m)$

Greedy algorithm:

for each $v \in V$
assign c_v different from
all c_u for $u \in N(u)$

always exists since have $\leq \Delta$ nbrs + palette size is $\Delta+1$

Can we do better?

Greedy list coloring:

Initial palette

For all v, let $\mathcal{L}(v) = \{1, .., \Delta+1\}$

For each $v \in V$ (arbitrary order)

if $\mathcal{L}(v) = \varphi$ output FAIL

else $c_v \leftarrow$ any color in $\mathcal{L}(v)$

remove c_v from all nbrs u of v

runtime:

$$\sum_v \Big(\underbrace{\text{time to find color in } \mathcal{L}(v) + \sum_{u \in N_v} \text{time to remove } c_v \text{ from } \mathcal{L}(u)}_{O(N_v)} \Big)$$

$= O(m)$

Sublinear Time Algorithm

↑ in m
not in n

Query model:

degree queries: what is $\deg(u)$?

pair queries: is $(u,v) \in E$?

nbr queries: what is k^{th} nbr of u?

Thm Can find $(\Delta+1)$-coloring in

$$\tilde{O}(n\sqrt{n}) \text{ time}$$

Comments
- no bound required on Δ for runtime
- non-adaptive
- $\Omega(n\sqrt{n})$ time required

Palette Sparsification

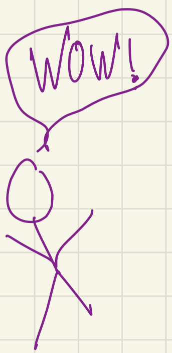
$\mathcal{L}(v)$ is v's sparser palette

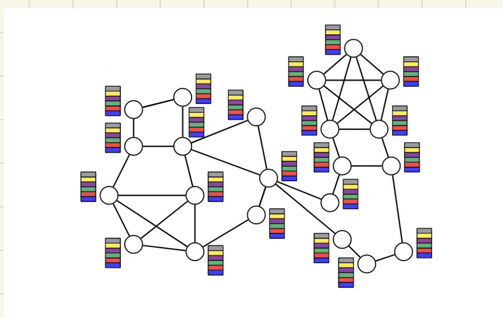
$\forall$ nodes v,

sample $k = O(\log n)$ colors $\mathcal{L}(v)$ from $\{1, \dots, \Delta+1\}$

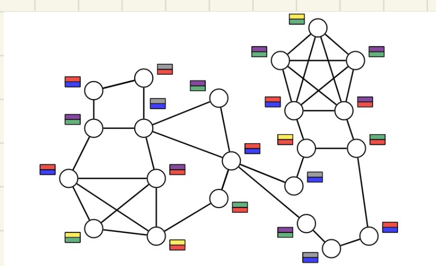
Surprisingly:
Sparsifying the palette this way (probably) doesn't kill colorability!

Main Claim whp, G can be colored s.t. $\forall v$, $c_v \in \mathcal{L}(v)$

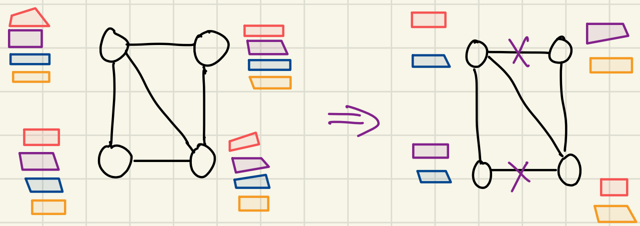




$\Downarrow$



simple argument that always exists a color left to use for v in greedy no longer holds!!



can ignore "x"ed edges since won't cause each other to lose colors from their palette

Why palette sparsification?

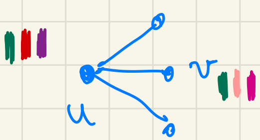
<u>idea</u> (since guaranteed that coloring respecting smaller palettes remains)

can $\underbrace{\text{throw out all } (u,v)}_{\text{sparsify edges in } G!!}$ s.t. $\underbrace{L(u) \cap L(v) \neq \emptyset}_{\text{they will not even consider using same colors!}}$

<u>how much sparser?</u>

whp $O(n \log^2 n)$ edges remain

<u>why</u>? $\forall u \in V$, let $c_1 \cdots c_K$ be colors chosen by u



$\forall v \in N(u)$ & $i \in [1..K]$

set $X_{v,i} \leftarrow \begin{cases} 1 & \text{if } v \text{ chooses color } c_i \\ 0 & \text{o.w.} \end{cases}$

(Should really be $X^{(u)}$ but notation is getting complicated)

Let $X \leftarrow \sum_{i=1}^{K} \underbrace{\sum_{v \in N(u)} X_{v,i}}_{\#\text{edges due to color } c_i}$ $\Big\{$ upper bnd on $\deg(u)$ in sparsified graph since v might share 2 colors with u

$$E[X_{v,i}] = \Pr[X_{v,i} = 1] = \frac{K}{\Delta+1}$$ $\leftarrow$ v chose K out of $\Delta+1$ colors too. what is probability it landed on i?

Then $$E[X] = \sum_{i=1}^{k} \sum_{v \in N(u)} E[X_{v,i}]$$ linearity of expectation

$$\leq K \cdot \Delta \cdot \frac{K}{\Delta+1} \leq K^2$$

Using $k = \Theta(\log n)$ shows $\forall u$, expected degree of u in remaining graph is $O(\log^2 n)$.

Can show whp with more work $\blacksquare$

palette sparsification $\implies$ sublinear time!

(also, good sublinear space "streaming" algorithms
+ massively parallel computation algorithms)

$\tilde{O}(n^2/\Delta)$-time Palette Coloring Algorithm

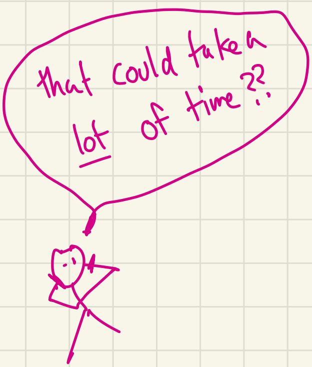
time

1. Construct palette $\forall\, u \in V$ — $O(n \log n)$
2. Construct G_{sparse}: — $O(n \log n)$

   $\forall c$, find $X_c = \{ v \mid c \in \mathcal{L}(v) \}$

if $u, v \in X_c$ + $(u,v) \in G$ then $(u,v) \in E_{sparse}$ $\longrightarrow$ query all pairs of nodes in each X_c to find E_{sparse}

how much time?



$$\le (\#\text{ colors } c) \times E\left[\binom{|X_c|}{2}\right] \le (\Delta+1)\, O\!\left(\frac{n^2 \log^2 n}{\Delta^2}\right)$$

$$E\left[\binom{|X_c|}{2}\right] \le E\Big[\sum_{\substack{u,v \\ \in V^2}} 1_{\substack{u+v \text{ pick} \\ \text{color } c}}\Big] = \binom{n}{2} \cdot \left(\frac{k}{\Delta+1}\right)^2 = O\!\left(\frac{n^2 \log^2 n}{\Delta^2}\right)$$

$$\le \tilde{O}\left(\frac{n^2}{\Delta}\right) \quad \text{(can be shown whp)}$$

3. perform greedy list coloring problem on G_{sparse}

recall Greedy list coloring: time: $O(|E_{sparse}|) = \tilde{O}(n)$

For each $v \in V$ (arbitrary order)
  if $L(v) = \phi$ output FAIL
  else $c_v \leftarrow$ any color in $L(v)$
    remove c_v from all nbrs u of v in G_{sparse}

<u>Corr</u> can find $\Delta+1$-coloring in $\tilde{O}(n^{3/2})$ time

<u>why</u>?

if $\Delta \le \sqrt{n}$

run greedy in $O(n\Delta) \le O(n\sqrt{n})$ time

if $\Delta > \sqrt{n}$

run palette sparsification + greedy list coloring:

$$\tilde{O}\left(\frac{n^2}{\Delta}\right) = \tilde{O}\left(\frac{n^2}{\sqrt{n}}\right) = \tilde{O}(n^{3/2})$$

time

$\blacksquare$

Why does main thm hold?

e.g. why is there still a $\Delta+1$ coloring after sparsification?

will show a weaker thm (allow 2Δ colors)

$\forall$ nodes v,
sample $O(\log n)$ colors $\mathcal{L}(v)$ from $\{1, \ldots, 2\Delta\}$

Weaker Claim whp, G can be colored s.t. $\forall v$, $c_v \in \mathcal{L}(v)$

Pf run Palette-coloring alg above with palette size s:
fail if a node ever runs out of colors

when attempt to color node v:

Gain: much weaker than saying no previous nbr had c in its list

say color $c \in \{1..2\Delta\}$ "good" if c not already used to color any nbr of v when v is considered

if $\mathcal{L}(v)$ contains any "good" color c, then can color v successfully

Since $\mathcal{L}(v)$ chosen independently of other lists, can think of choosing it "now". Since v has $\leq \Delta$ nbrs,

$$\Pr[\mathcal{L}(v) \text{ contains no } \underbrace{\text{good color}}_{\text{all } c \text{ are "bad"}}]$$

$$\leq \Pr[\text{pick all colors of } v \text{ from colors already chosen by nbrs}]$$

$$\leq \frac{\binom{\Delta}{s}}{\binom{2\Delta}{s}} = \frac{\frac{(\Delta)(\Delta-1)\cdots(\Delta-s+1)}{s!}}{\frac{(2\Delta)(2\Delta-1)\cdots(2\Delta-s+1)}{s!}}$$

$$< \frac{1}{2^s} = \frac{1}{n^c}$$

Choosing $s = c\log n$

+ union bnd for all $v \in V$

$\Rightarrow$ whp algorithm never FAILS

$\Rightarrow$ whp G has legal list coloring

$\blacksquare$