Lecture 5:

- Greedy algorithms vs. Sublinear time:

  the case of maximal matching

- Property Testing:

  is the graph planar?

Sublinear time algorithms via greedy:

We focus on problem of

estimating size of maximal matching (MM)
in degree bounded graph

why?
- step towards approx maximum matching
- relation to Vertex cover (VC)

$VC \geq MM$ ← for each edge in matching, $\underbrace{\geq 1 \text{ endpt}}_{\text{these are disjoint!}}$ must be in VC

$VC \leq 2 \cdot MM$ ← put all MM nodes in VC
if any edge not covered by VC
then can add edge to MM
violating maximality of MM.

Note (similar to VC)

if degree $\leq \Delta$, maximal matching $\geq \frac{m}{2\Delta}$

why? run process:

place edge (u,v) in MM

delete other edges of u + V ($\leq 2\Delta$) which can no longer be in matching

Greedy Sequential Matching Algorithm:

$M \leftarrow \emptyset$

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

if neither u or v matched

add e to M

Output M

Observation:

M is maximal

why? if $\underset{(u,v)}{\underset{\shortparallel}{e}} \notin M$ either u or V already matched earlier

Oracle Reduction Framework:

Assume given deterministic "oracle" $\mathcal{O}(e)$
which tells you if $e \in M$ or not in one step

Algorithm to estimate $|M|$:

- $S' \leftarrow$ set of $s = \frac{8}{\varepsilon^2}$ nodes chosen iid
- $\forall v \in S'$
  let $X_v \leftarrow \begin{cases} 1 & \text{if any call to } \mathcal{O}(v,w) \text{ for } w \in N(v) \text{ returns "yes"} \\ 0 & \text{o.w.} \end{cases}$
- Output $\underbrace{\frac{n}{2s} \sum_{v \in S'} X_v}_{\text{since 2 nodes matched for each edge in } M} + \underbrace{\frac{\varepsilon}{2} \cdot n}_{\text{makes underestimate unlikely}}$

Behavior of output:

(why a good approximation?)

- $S' \leftarrow$ set of $s = \frac{8}{\varepsilon^2}$ nodes chosen iid
- $\forall v \in S'$
  let $X_v \leftarrow \begin{cases} 1 & \text{if any call to } \mathcal{O}(v,w) \text{ for } w \in N(v) \text{ returns "yes"} \\ 0 & \text{o.w.} \end{cases}$
- Output $\frac{n}{2s} \sum_{v \in S'} X_v + \frac{\varepsilon}{2} \cdot n$

note $|M| = \frac{1}{2} \sum_{v \in V} X_v$

$$E[\text{output}] = E\left[\frac{n}{2s} \sum_{v \in S} X_v\right] + \frac{\varepsilon}{2} \cdot n$$

$$= \frac{n}{2s} \sum_{v \in S} E[X_v] + \frac{\varepsilon}{2} \cdot n$$

but $E[X_v] = \underbrace{\frac{2|M|}{n}}_{\text{fraction of matched nodes}}$

$$= \frac{\cancel{n}}{\cancel{2s}} \cdot \cancel{s} \cdot \cancel{2} \frac{|M|}{\cancel{n}} + \frac{\varepsilon}{2} \cdot n$$

$$= |M| + \frac{\varepsilon}{2} n$$

$$\Pr\left[\left|\left(\frac{n}{2s} \sum_{v \in S} X_v + \frac{\varepsilon}{2} \cdot n\right) - E[\text{output}]\right| \geq \frac{\varepsilon}{2} \cdot n\right]$$

$$\parallel$$

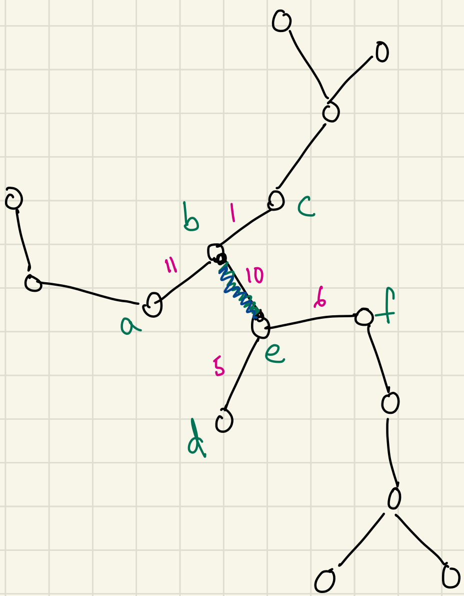
$$\Pr\left[\left|\frac{n}{2s} \sum_{v \in S} X_v - |M|\right| \geq \frac{\varepsilon}{2} \cdot n\right] \leq \frac{1}{3}$$ by additive Chernoff-Hoeffding

Claim with prob $\geq 2/3$, $|M| \leq \text{output} \leq |M| + \varepsilon n$

Implementing the oracle:

Main idea: figure out "what would $\underbrace{\text{greedy}}$ do on (v,w)?"

how?
which input order?
do we need to figure out <u>all</u> previous nodes?



Is $(b,e) \in M$?

adjacent to

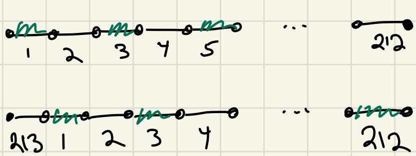
$$\underbrace{\underset{1}{(b,c)}} \quad \underset{5}{(e,d)} \quad \underset{6}{(e,f)} \quad \underset{11}{(a,b)}$$

greedy considers 1st
& puts (b,c) into M
so $(b,e) \notin M$

$\Rightarrow$ no need to consider rest of graph

Problem: Greedy is "sequential" & has long dependency chains?

example:



even if you know graph is line, is edge odd or even in order?

Implementation of oracle:

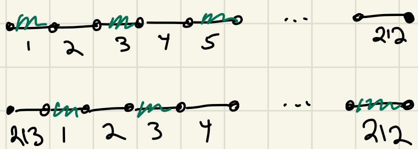
Input: edge e
Output: is $e \in M$?

Algorithm:

- recursively find all decisions for adjacent edges with lower ordering number (do not need info on adjacent edges with higher number since greedy doesnt consider before e)
- if any adj. edge with lower number is matched then e is not matched
  else e is matched

Problem: Greedy is "sequential" + has long dependency chains?

example:

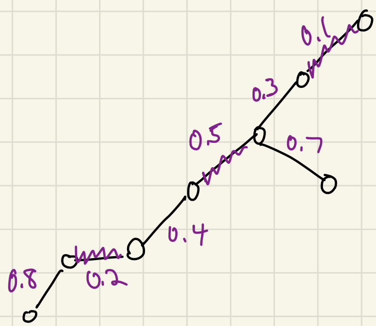


even if you know graph is line, is edge odd or even in order?

How to break length of dependency chains?

assign random ordering to edges

Example:



Is edge 0.5 in M?

- recurse on 0.3
  - recurse on 0.1
    - no other adjacent edges so 0.1 matched
    - therefore 0.3 not matched
- no need to recurse on 0.7 since 0.5 < 0.7
- recurse on 0.4
  - recurse on 0.2
    - 0.8 comes after 0.2
    - 0.4 " " "
    - so 0.2 matched
    - so 0.4 not matched
- 0.5 matched

Implementation of oracle:

assume ranks r_e assigned to each edge e

to check if $e \in M$:

$\forall e'$ neighboring e,

- if $r_{e'} < r_e$ recursively check e'

  & if $e' \in M$, return "$e \notin M$" & halt

  else continue

return "$e \in M$"

↑ since no e' of lower rank than e is in M

Correctness:

follows from correctness of greedy

Query complexity:

Claim expected # queries to graph per oracle query is $2^{O(d)}$

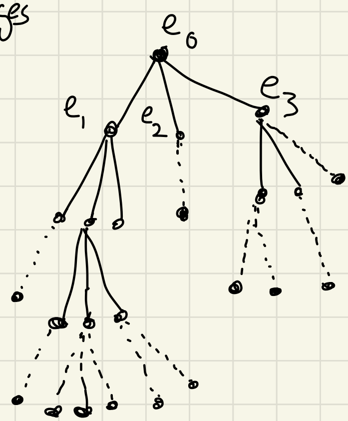
Claim + Parnas-Ron reduction $\Rightarrow$ total query complexity is $\frac{2^{O(d)}}{\varepsilon^2}$

$\forall e'$ neighboring e,
- if $r_{e'} < r_e$ recursively check e'
  - & if $e' \in M$, return "$e \notin M$" & halt
- else continue

return "$e \in M$"

Pf of Claim:

- Consider query tree:
  root node labelled by original queryedge
  children of each node are adjacent edges



- will only query paths that are monotone decreasing in rank

- $\Pr[\text{given path of length } k \text{ explored}] = \frac{1}{(k+1)!}$

- # edges in original graph at dist $=k$ in tree is at most d^k

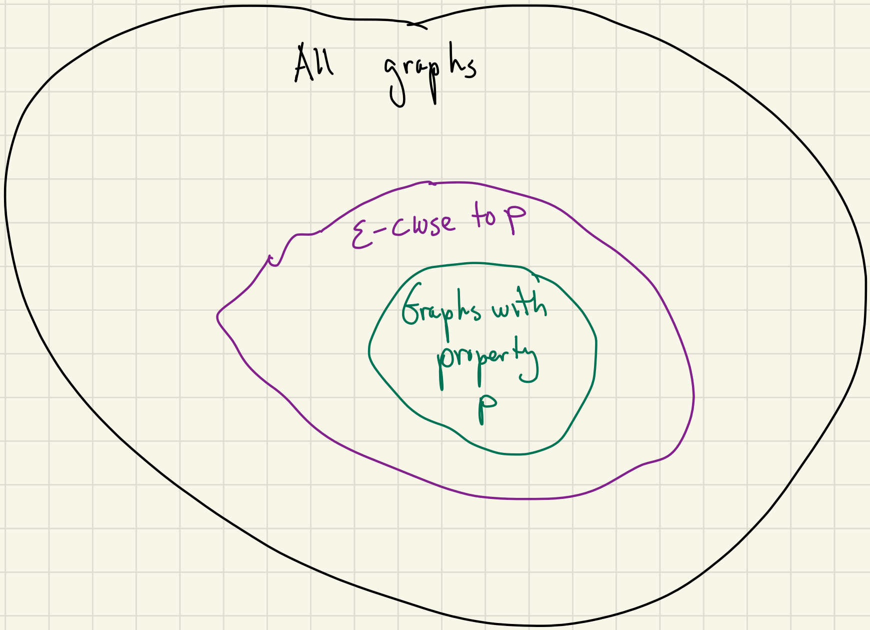
- $E[\text{\# edges explored at dist} = k] \le \frac{d^k}{(k+1)!}$

- $E[\text{total \# edges explored}] \le \sum_{k=0}^{\infty} \frac{d^k}{(k+1)!}$

$\le \frac{e^d}{d}$

$\blacksquare$

Property Testing



examples of P:
- planar
- bipartite
- no small cuts
- no triangles
- Connected

Can we distinguish graphs with property P from far from P?

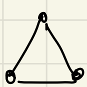
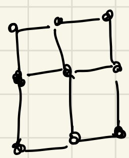
e.g. G is ε-far from planar if must remove $\geq \varepsilon \cdot \Delta \cdot n$ edges to make it planar

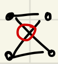
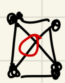
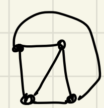
Today + next time:

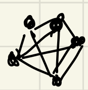
test planarity in time independent of n
(but exponential in ε)

for graphs with max degree Δ

what is a planar graph?
Can be drawn on plane s.t. edges don't intersect

K_3   Yes

$K_{2,2}$  K_4  NO? actually, yes  

$K_{3,3}$  K_5  NO!

Cool characterization of planar graphs:

def. H is "minor" of G if can obtain H from G via vertex removals, edge removals or edge contractions

Minor closed properties:

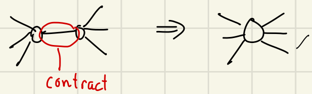
Let P be a set of graphs

e.g. P = planar graphs ← minor-closed
P = bipartite graphs ← not minor-closed
⋮

P is "minor closed" if $\forall G \in P$ then all minors of G are in P

Minor free graph families:

def. H is "minor" of G if can obtain H from G via vertex removals, edge removals or edge contractions



def G is "H-minor-free" if H not a minor of G

Thm [Kuratowski]

G is planar iff G is $K_{3,3}$ + K_5 minor free

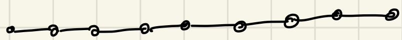
another example: bounded tree width (there are more...)

Really cool theorem: [Robertson & Seymour]

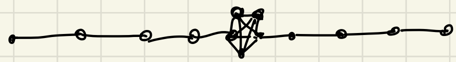
Every minor-closed property is expressible as a constant # of excluded minors.

Testing Planarity:

Can't hope to distinguish



from



in sublinear time!

def G is "ε- close to H-minor-free"

if can remove $\leq \varepsilon \cdot \Delta \cdot n$ edges to

make it H - minor free

Specifically:

def G is ε-close to planar iff

can remove $\leq \varepsilon \cdot \Delta n$ edges to make it

$\left\{ \begin{array}{l} \text{planar} \\ K_{3,3} + K_5\text{- free} \end{array} \right.$ $\Leftrightarrow$ equivalent

else G is ε-far

Goal: Given G

- if G planar, PASS
- if G ε-far from planar, FAIL

$\Big\}$ with prob $\underbrace{\geq 2/3}_{\text{arbitrary const} \geq 1/2}$

Plan for tester: use nice property of Planar (all all H-minor free) graph families.

Can always remove $\underbrace{\text{small fraction}}_{\leq \varepsilon}$ of edges

& break up graph into $\underbrace{\text{tiny}}_{\leq \text{const}}$ connected components

def. G is "(ε, k)-hyperfinite" if

Can remove $\leq \varepsilon n$ edges &

remain with connected components of

size $\leq k$

Useful Thm

Given H, $\exists$ const C_H s.t.

$\forall\ 0 < \varepsilon < 1$, every H-minor-free graph G

of deg $\leq \Delta$ is $\left(\underbrace{\varepsilon \cdot \Delta}_{\substack{\text{remove} \\ \leq \varepsilon \cdot \Delta \cdot n \\ \text{edges}}}, \underbrace{\frac{C_H}{\varepsilon^2}}_{\substack{\text{Components of size } O\left(\frac{1}{\varepsilon^2}\right) \\ \text{no dependence on } n}}\right)$-hyperfinite

<u>note</u> subgraphs of H-minor free graphs are also H-minor free, so <u>also</u> hyperfinite

but only remove # edges in proportion to # nodes in subgraph

$\Rightarrow$ can recurse & break up further]