Lecture 5

- a fast distributed algorithm for VC (notes from last lecture)
- Estimating the size of a maximal matching
  via simulating greedy algorithm
- Testing H-minor freeness -
  in particular, testing planarity

Sublinear Time Approximation Algorithms:

Estimating size of maximal matching in degree bounded graph

Why?

- relation to Vertex Cover
  - $VC \geq MM$ ← for each edge in matching, ≥1 endpt must be in VC (these are disjoint)
  - $VC \leq 2MM$ ← put all MM nodes in VC; if an edge not covered, then violates maximality
- a step towards approx maximum matching

Note: if deg $\leq d$, Maximal matching $\geq \frac{n}{d}$ ← to see this, run greedy algorithm

Greedy Sequential Matching Algorithm:

$M \leftarrow \phi$

$\forall\ e = (u,v) \in E$,
  if neither u or v matched,
    add e to M

Output M

} output depends only on ordering of input edges

Observe:
M maximal, since if $e \notin M$ (i.e. (u,v)) either u or v already matched earlier

Implementing the oracle:

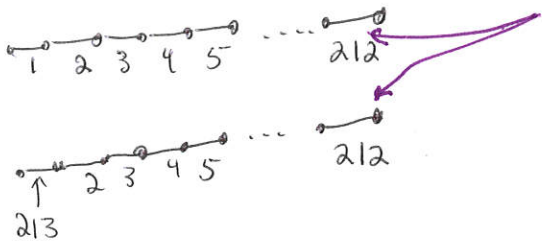
Main idea: figure out "what would greedy do on (π, w)?"

how? according to which input order?

Problem: greedy is "sequential"

Can have long dependency chains

example:



even if you know the graph is a line, how do you know if edge is odd or even in the order?

How to implement oracle based on greedy?

To decide if e in matching,

- need to know decisions for **adjacent** edges that came **before** e in ordering
- do not need to know anything about any edge that comes **after** e in ordering since not considered by greedy algorithm before e

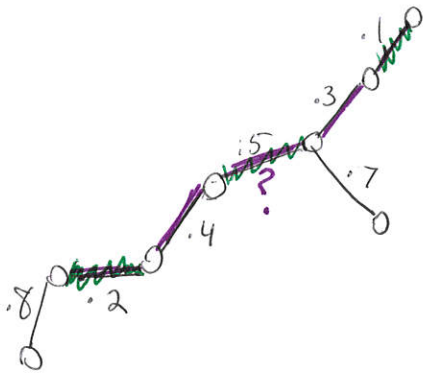
so, if any adjacent edge before e in ordering matched, e is not matched

otherwise e is matched

How to break length of dependency chains?

assign random ordering to edges

example



is edge .5 in M?

- recurse on .3
  - recurse on .1
    - no other adjacent edges so .1 is matched
    - therefore .3 is not matched
    - no need to recurse on .7
  - don't know yet about .5 so recurse on .4
    - recurse on .2
      - .8 comes after .2 in order so doesn't affect Greedy's behavior
      - same for .4
      - so .2 is matched
    - .4 is not matched
  - .5 is matched

Implementation of oracle: assume ranks r_e assign 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

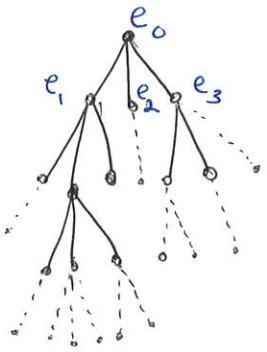
Query complexity:

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

Claim $\Rightarrow$ total query complexity is $\frac{2^{O(d)}}{\varepsilon^2}$

Pf of Claim

- Consider Query Tree where root node labelled by original query edge, children of each node are edges adjacent to it.



- 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 $\leq k$ in tree $\leq d^k$

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

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

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

- $E[\text{query complexity}] \leq d \cdot \frac{e^d}{d} = e^d = 2^{O(d)}$

▨

Testing H-minor freeness

all graphs have max degree $\leq d$

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

• G is "H-minor-free" if H not minor of G

• G is "ε-close to H-minor-free" if
can remove $\leq \varepsilon dn$ edges to make it
H-minor-free

• minor closed property P –
if $G \in P$ then all minors of G are in P

Really Cool Theorem [Robertson + Seymour]
Every minor-closed property is expressible
as a constant # of excluded minors.

Some minor-closed properties:
planar graph, bounded tree width, ...

Goal: Testing H-minor freeness
Pass H-minor free graphs
Fail if far from H-minor free

more definitions

- G is "(ε, k)-hyperfinite" if

  Can remove $\leq \varepsilon n$ edges

  & remain with connected components of size $\leq k$

- G is "ρ-hyperfinite" if

  $\forall\ \varepsilon > 0$, G is $(\varepsilon, \rho(\varepsilon))$-hyperfinite

Useful Thm

Given H

$\exists\ C_H$ s.t. $\forall\ 0 < \varepsilon < 1$, every H-minor free graph of deg $\leq d$

(constant that depends only on H)

is $(\varepsilon d, C_H^2/\varepsilon^2)$- hyperfinite.

(ie. remove $\leq \varepsilon dn$ edges & components of size $O(1/\varepsilon^2)$)

note

Subgraphs of H-minor free graphs also H-minor free

& so also hyperfinite

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