Lecture 9:

S-t connectivity in deterministic logspace

Undirected s-t connectivity revisited

given: undir G
nodes s, t

Question: are s, t in same component?

an easy case:

def: $(\underset{\uparrow\ \text{\#nodes}}{N}, \underset{\nwarrow\ \text{degree}}{D}, \underset{\nwarrow\ \text{upper bnd on } \lambda_2 \text{ of transition matrix}}{\lambda})$ - graph

a well-known-fact: Tanner, Alon-Milman

$\forall\ \lambda < 1,\ \exists\ \varepsilon > 0$ s.t. $\forall\ (N, D, \lambda)$ - graphs G

$+ \forall\ S$ s.t. $|S| < \frac{N}{2}$

$$\left.\begin{array}{l} |\underbrace{N(S)}_{\text{includes } S}| \geq (1+\varepsilon)\,|S| \end{array}\right\} \text{ i.e. } G \text{ "expands"}$$

fact implies another easy fact: such a G also has $O(\log N)$ diameter

Idea for algorithm in which each component of graph is (N, D, λ) for $\lambda < 1$ +const D (or just $\log n$ - diameter)

- enumerate all D^{ℓ} paths (for $\ell = O(\log N)$) starting at s
- if ever see t, output "connected"

Space requirements:

assume lexicographic ordering on paths (comes from ordering on outedges)

just keep track of DFS path from s:

- const # bits per step of path
- O(log n) length

Total: O(log n) bits

Behavior:

if s,t not connected, never answers connected

if s,t connected - will find path

Problem: not all graphs are
- (N, D, λ)-graphs for λ<1
- or even just O(log n) diameter

In general, we know:

Thm ∀ connected non-bipartite graphs, $\lambda(G) \le 1 - \frac{1}{\underbrace{DN^2}_{\text{not too good!}}}$

What about powering?

if G is (N, D, λ) then G^t is (N, D^t, λ^t)

good or bad?

⊕ reduce 2nd e-val

⊖ increase degree

Need operation which reduces degree w/o killing 2nd e-val

i.e. 1) if it was expander, should remain so but reduce degree

2) don't need to increase expansion, powering will do that

Lets start with a "base graph"

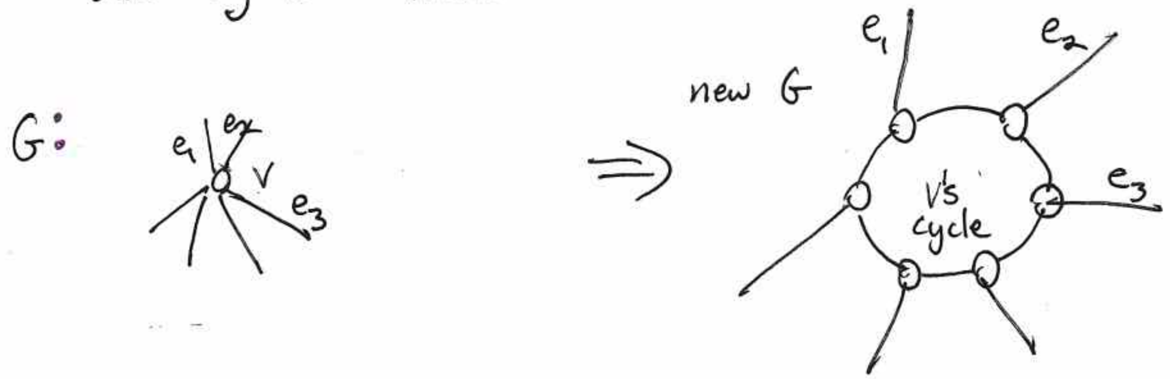
Thm 1 $\exists$ const D_e + $\Big(\underset{\overset{\uparrow}{N}}{(D_e)^{16}},\ \underset{\overset{\uparrow}{D}}{D_e},\ \underset{\overset{\uparrow}{\lambda}}{\tfrac{1}{2}}\Big)$-graph

Constant size - can find this by enumeration

A first issue to handle:

nice to have regular graph of const degree d with same connected components

one way to transform G:



- quadratic blowup in # of nodes, but just once
- can add self loops to deg < d nodes

in both cases, easy to fix neighbor fctn in log space

Could be bad for λ but we'll fix later...

A second issue: representing graphs

adjacency lists?

Rotation map $Rot_G : [N] \times [D] \to [N] \times [D]$

$Rot_G(v,i) = (w,j)$ if i^{th} edge of v leads to w & j^{th} edge of w leads to v

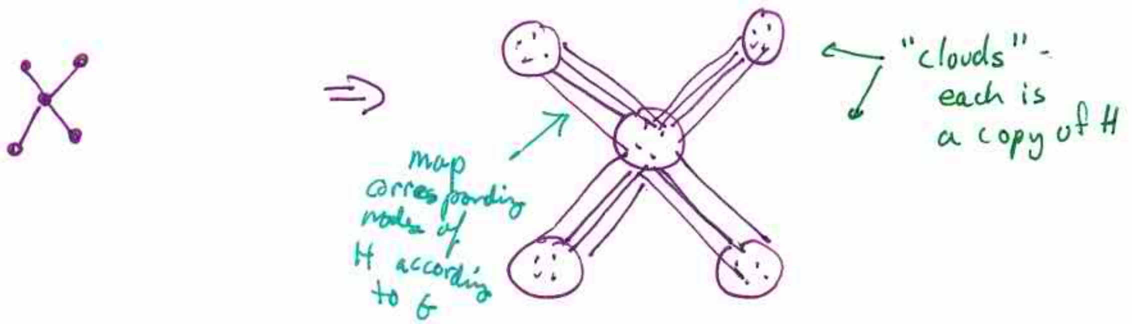
allows back & forth on same edge!

① Replacement Product $G \textcircled{r} H$

Given G D-regular N nodes, H d-regular D nodes $\Big\}$ G' with $N \cdot D$ nodes, degree $d+1 << D$

nodes: $v \in G$ replaced by copy of H

edges: each vertex in H_v connected to nbrs in H_v
if u is i^{th} nbr of v in G + v is u's j^{th} nbr
add edge from i^{th} node of H_v to j^{th} node of H_u



map corresponding nodes of H according to G

"clouds" - each is a copy of H

② Zig-zag product $G \textcircled{z} H$

Given G D-regular, N nodes, H d-regular D nodes $\Big\}$ G'' with $N \cdot D$ nodes, degree d^2

nodes: as in G', $v \in G$ replaced by copy of H

edges: path of length 3 in G' i.e. $(u,v) \in G''$ iff $u \in H_i$ ("cloud i")

$$\left.\underset{\overset{\uparrow}{d \text{ choices}}}{(u,w) \in E(H_i)},\ \underset{\overset{\text{st. } z \in H_j}{\overset{\uparrow}{1 \text{ choice}}}}{(w,z) \in E(G)},\ \underset{\overset{\uparrow}{d \text{ choices}}}{(z,v) \in E(H_j)}\ \right\} \text{degree } d^2$$

Some intuition:

in terms of cuts -

to find min cut, would want to
break s.t. clouds intact (since clouds are expanders)
$\Rightarrow$ relative cut size should be similar to G's

in terms of random walks -

two extreme cases:

1) distribution far from uniform in each cloud:
then walks on H make it more uniform
& G step won't affect

2) uniform within clouds but different weights
on clouds:
then walks on H won't affect,
& walking on G improves slowly

<u>Thm</u> For $\alpha \le 1/2$ For G an (N, D, λ)-graph + H a (D, d, α)-graph, $G \textcircled{z} H$ is $(ND, d^2, \lambda_{G \textcircled{z} H})$-graph
s.t. $\frac{1}{2}(1-\alpha^2)(1-\lambda) \le 1 - \lambda_{G \textcircled{z} H}$

So,

$$\lambda_{G z H} \le 1 - \frac{1}{2}(1 - \underbrace{\overset{\le 1/2}{\alpha^2}}_{}) (1-\lambda)$$

$$\lambda_{G z H} \le 1 - \frac{1}{2}\underbrace{(1-\alpha^2)}_{\ge 3/4}(1-\lambda)$$

$$\le 1 - 3/8(1-\lambda)$$

$$\le 1 - \frac{1}{3}(1-\lambda)$$

$$\le 2/3 + \lambda/3 \leftarrow \text{still } < 1\text{, now, let's power it up a few times!}$$

How do we use this?

Main Transformation:

Given: G D^{16}-regular on N nodes
H D-regular on D^{16} nodes (Thm 1)

Transformation:

$\ell \leftarrow$ smallest int s.t. $\left(1 - \frac{1}{DN^2}\right)^{2^\ell} < 1/2$

$G_0 \leftarrow G$

$$G_i \leftarrow \underbrace{(\underset{\uparrow\ \text{degree reduction}}{G_{i-1} \textcircled{z} H})^{8}}_{} \nwarrow \text{powering}$$

Output: G_ℓ

What are properties of G_ℓ ?

\# nodes = $N \cdot (D^8)^\ell$ which is poly(N) since

$\ell = O(\log N)$

$+ D = O(1)$

Lemma if $\lambda(H) \le 1/2$ + G connected, not bipartite

then $\lambda(G_\ell) \le 1/2$

Pf idea $\lambda_{G_0} \le 1 - \frac{1}{DN^2}$

need Claim $\lambda_{G_i} \le \max\{\lambda(G_{i-1})^2, \frac{1}{2}\}$

if Claim, then for $d = \Theta(\log DN)$.

$$\text{have} \quad \lambda(G_\ell) \le \max\{\lambda(G_0)^{2^\ell}, \tfrac{1}{2}\}$$

$$\le \max\left\{\left(1 - \tfrac{1}{DN^2}\right)^{c \cdot DN^2}, \tfrac{1}{2}\right\} \le 1/2$$

Then Prove claim by inductionon i.

▨

Final Algorithm

1. preprocess G to make regular, nonbipartite
with same connected components
(can do by $G \textcircled{r}$ N-cycle ~~&~~ then add self-loops)
new graph has N^2 nodes

2. use zigzag + powering transformation to get G_ℓ

3. run expander algorithm on G_ℓ

A final issue:

how do you perform walks in logspace?

need to show that can compute rotation map of G_ℓ given rot map of G, H

use rotation maps!
gives way of going backwards & forwards on a path
$\Rightarrow$ only need to remember a constant # of bits to choose next step of path