This talk is about *spanners*

Given graph \( G = (V, E) \), subgraph \( H \) of \( G \) is a *\( t \)-spanner* of \( G \) if

\[
d_H(u, v) \leq t \cdot d_G(u, v) \quad \text{for all } u, v \in V
\]

- \( t \) is the *stretch* of the spanner.
- In this paper: \( G \) undirected, unweighted, connected
- Sufficient for stretch condition to hold for all edges \( \{u, v\} \in E \)
Graph Spanners: Basics

This talk is about spanners

Given graph $G = (V, E)$, subgraph $H$ of $G$ is a $t$-spanner of $G$ if

$$d_H(u, v) \leq t \cdot d_G(u, v) \quad \text{for all } u, v \in V$$

- $t$ is the stretch of the spanner.
- In this paper: $G$ undirected, unweighted, connected
- Sufficient for stretch condition to hold for all edges $\{u, v\} \in E$
Classical Objectives

Want to have small stretch, and small “cost”.
Two natural cost measures: total # edges, maximum degree.
Classical Objectives

Want to have small stretch, and small “cost”.
Two natural cost measures: total \# edges, maximum degree.

\# edges: [Althöfer et al ’93]:
- For any positive integer \( k \), all graphs have a \((2k - 1)\)-spanner with \( O(n^{1+1/k}) \) edges, and
Classical Objectives

Want to have small stretch, and small "cost".
Two natural cost measures: total \# edges, maximum degree.

**\# edges: [Althöfer et al '93]:**
- For any positive integer $k$, all graphs have a $(2k - 1)$-spanner with $O(n^{1+1/k})$ edges, and
- There exist graphs in which all $(2k - 1)$-spanners have $\Omega(n^{1+1/k})$ edges (assuming Erdős Girth Conjecture).
Classical Objectives

Want to have small stretch, and small “cost”.
Two natural cost measures: total \# edges, maximum degree.

\# edges: [Althöfer et al ’93]:
- For any positive integer $k$, all graphs have a $(2k - 1)$-spanner with $O(n^{1+1/k})$ edges, and
- There exist graphs in which all $(2k - 1)$-spanners have $\Omega(n^{1+1/k})$ edges (assuming Erdös Girth Conjecture).

No such theorem possible for max degree! Star graph. Removing any edge cases infinite stretch
Switch our point of view from tradeoffs to optimization.

Given $G, k$, efficient algorithm for finding best $t$-spanner of $G$?
Switch our point of view from tradeoffs to optimization.

Given $G, k$, efficient algorithm for finding best $t$-spanner of $G$?

- **Basic $t$-Spanner**: “best” = fewest edges
  - Lots known – come chat with me!
  - High-level view: can’t really beat trivial $O(n^{1/k})$-approximation for $t = 2k - 1$.
  - Can slightly in some special cases: $t = 3$ [BBMRY ’13] and $t = 4$ [D-Zhang ’16]

- **Lowest Degree $t$-Spanner (LD$t$S)**: “best” = min max degree
  - Chlamtác-D ’16: $O(\Delta^{(1-\frac{1}{t})^2})$-approximation, $\Omega(\Delta^{1/t})$ lower bound
  - Chlamtác-D-Krauthgamer ’12: $\tilde{O}(\Delta^{3-2\sqrt{2}})$-approx when $t = 2$ (Sherali-Adams)
Classical Objectives: Motivation and Issues

- **Number of Edges:**
  - **Pros:** natural objective, very nice tradeoff theorems. Well-studied. Often what’s needed in applications.
  - **Cons:** Do we really not care if one node has huge degree, as long as others small? Load in distributed settings?

- **Maximum Degree:**
  - **Pros:** Encourages low loads in distributed settings. Natural objective.
  - **Cons:** If some node forced to have large degree, do we really want to allow all other nodes to have large degree?
Classical Objectives: Motivation and Issues

- **Number of Edges:**
  - Pros: natural objective, very nice tradeoff theorems. Well-studied. Often what’s needed in applications.
  - Cons: Do we really not care if one node has huge degree, as long as others small? Load in distributed settings?

- **Maximum Degree:**
  - Pros: Encourages low loads in distributed settings. Natural objective.
  - Cons: If some node forced to have large degree, do we really want to allow all other nodes to have large degree?

- Want something new: encourages max degree to be small, but also encourages other nodes to have small degree even if max forced to be large.
New Objective

- Observation: consider vector \( d_G \in \mathbb{Z}_{\geq 0}^n \) of vertex degrees in \( G \).
  - Number of edges is \( \frac{1}{2} \|d_G\|_1 \)
  - Maximum degree is \( \|d_G\|_\infty \)
- Interpolate between the two!
New Objective

- Observation: consider vector $d_G \in \mathbb{Z}_{\geq 0}^n$ of vertex degrees in $G$.
  - Number of edges is $\frac{1}{2} \|d_G\|_1$
  - Maximum degree is $\|d_G\|_\infty$
- Interpolate between the two!

The $\ell_p$-norm objective is to minimize

$$\|H\|_p = \|d_H\|_p = \left( \sum_{u \in V} d_H(u)^p \right)^{1/p}$$

- For $1 < p < \infty$, encourages both sparsity and low maximum degree!
  - Standard objective in clustering, scheduling, etc.
\(\ell_p\)-Objective: Tradeoffs

Introduced this objective in [Chlamtáč-D-Robinson ICALP ’19]

**Theorem:** For every \(k, p \geq 1\), every graph admits a \((2k - 1)\)-spanner with \(\ell_p\)-norm max(\(O(n)\), \(O(n^{\frac{k+p}{kp}})\)). This bound is also tight.
\(\ell_p\)-Objective: Tradeoffs

Introduced this objective in [Chlamtáč-D-Robinson ICALP ’19]

**Theorem:** For every \(k, p \geq 1\), every graph admits a \((2k - 1)\)-spanner with \(\ell_p\)-norm
\[
\max(O(n), O(n^{\frac{k+p}{kp}})).
\]
This bound is also tight.

Solved the tradeoff question, but what about optimization?

**Definition:** In the **Minimum \(\ell_p\)-Norm t-Spanner** problem, we are given \(p, t, G\), and our goal is to find the \(t\)-spanner \(H\) of \(G\) minimizing \(\|H\|_p\)

Focus of this paper, with \(p = 2, t = 3\) (some results generalizable)
Results

First, study greedy algorithm (used to prove tradeoffs).

Greedy is an $\tilde{O}(n^{3/7})$-approximation for \textsc{Minimum $\ell_2$-Norm 3-Spanner} (and this is tight).

New algorithm based on rounding convex relaxation.

There is an $\tilde{O}(n^{5/13})$-approximation for \textsc{Minimum $\ell_2$-Norm 3-Spanner}.

Hardness result (more careful analysis of max-degree hardness).

Unless $\textbf{NP} \subseteq \textbf{BPTIME}(2^{\text{polylog}(n)})$, for any $\epsilon > 0$ there is no polynomial-time algorithm for \textsc{Minimum $\ell_2$-Norm 3-Spanner} with approximation ratio better than $2^{\log^{1-\epsilon} n}$. 
Why greedy?

Since we have a better algorithm, why study greedy?

\[ \ell_2 \text{ norm for stretch } 3 \text{ is fundamentally different from } \ell_1 \text{ or } \ell_\infty \text{ in terms of greedy approximation.} \]

\( \ell_1 \): Greedy always has at most \( O \left( \frac{n^3}{2} \right) \) edges, so trivially an \( O \left( \frac{n^1}{2} \right) \)-approximation. Tight.

\( \ell_\infty \): Greedy has max degree at most \( \Delta \), and \( \text{OPT} \geq \frac{\Delta^1}{3} \). So \( O \left( \frac{\Delta^2}{3} \right) = O \left( \frac{n^2}{3} \right) \)-approximation. Tight.

\( \ell_2 \): Greedy has at most \( O(n) \) edges (tight), \( \text{OPT} \geq \Omega \left( \frac{n^1}{2} \right) \) (tight). But greedy is \( \tilde{O} \left( \frac{n^3}{7} \right) \)-approximation!

Approximation ratio of greedy cannot be determined by "absolute" guarantees for \( p = 2 \), unlike \( p = 1, \infty \)!

Interesting analysis: write a constant-size LP, argue it characterizes approximation ratio, give tight bound on LP.
Why greedy?

Since we have a better algorithm, why study greedy?

- Natural and important algorithm – should understand its performance!
- Demonstrates that $\ell_2$ norm for stretch 3 is fundamentally different from $\ell_1$ or $\ell_\infty$ in terms of greedy approximation.
Why greedy?

Since we have a better algorithm, why study greedy?

- Natural and important algorithm – should understand its performance!
- Demonstrates that $\ell_2$ norm for stretch 3 is fundamentally different from $\ell_1$ or $\ell_\infty$ in terms of greedy approximation.
  - $\ell_1$: Greedy always has at most $O(n^{3/2})$ edges, so trivially an $O(n^{1/2})$-approximation. Tight.
  - $\ell_\infty$: Greedy has max degree at most $\Delta$, and $OPT \geq \Delta^{1/3}$. So $O(\Delta^{2/3}) = O(n^{2/3})$-approximation. Tight.
Why greedy?

Since we have a better algorithm, why study greedy?

- Natural and important algorithm – should understand its performance!
- Demonstrates that $\ell_2$ norm for stretch 3 is **fundamentally different** from $\ell_1$ or $\ell_\infty$ in terms of greedy approximation.
  - $\ell_1$: Greedy always has at most $O(n^{3/2})$ edges, so trivially an $O(n^{1/2})$-approximation. Tight.
  - $\ell_\infty$: Greedy has max degree at most $\Delta$, and $\text{OPT} \geq \Delta^{1/3}$. So $O(\Delta^{2/3}) = O(n^{2/3})$-approximation. Tight.
  - $\ell_2$: Greedy has at most $O(n)$ edges (tight), $\text{OPT} \geq \Omega(n^{1/2})$ (tight). But greedy is $\tilde{O}(n^{3/7})$-approximation!
Why greedy?

Since we have a better algorithm, why study greedy?

- Natural and important algorithm – should understand its performance!
- Demonstrates that $\ell_2$ norm for stretch 3 is fundamentally different from $\ell_1$ or $\ell_\infty$ in terms of greedy approximation.
  - $\ell_1$: Greedy always has at most $O(n^{3/2})$ edges, so trivially an $O(n^{1/2})$-approximation. Tight.
  - $\ell_\infty$: Greedy has max degree at most $\Delta$, and $OPT \geq \Delta^{1/3}$. So $O(\Delta^{2/3}) = O(n^{2/3})$-approximation. Tight.
  - $\ell_2$: Greedy has at most $O(n)$ edges (tight), $OPT \geq \Omega(n^{1/2})$ (tight). But greedy is $\tilde{O}(n^{3/7})$-approximation!
- Approximation ratio of greedy cannot be determined by “absolute” guarantees for $p = 2$, unlike $p = 1, \infty$!
Why greedy?

Since we have a better algorithm, why study greedy?

- Natural and important algorithm – should understand its performance!
- Demonstrates that $\ell_2$ norm for stretch 3 is fundamentally different from $\ell_1$ or $\ell_\infty$ in terms of greedy approximation.
  - $\ell_1$: Greedy always has at most $O(n^{3/2})$ edges, so trivially an $O(n^{1/2})$-approximation. Tight.
  - $\ell_\infty$: Greedy has max degree at most $\Delta$, and $\text{OPT} \geq \Delta^{1/3}$. So $O(\Delta^{2/3}) = O(n^{2/3})$-approximation. Tight.
  - $\ell_2$: Greedy has at most $O(n)$ edges (tight), $\text{OPT} \geq \Omega(n^{1/2})$ (tight). But greedy is $\tilde{O}(n^{3/7})$-approximation!

- Approximation ratio of greedy cannot be determined by “absolute” guarantees for $p = 2$, unlike $p = 1, \infty$!
- Interesting analysis: write a constant-size LP, argue it characterizes approximation ratio, give tight bound on LP.
Approximation Algorithm: Convex Relaxation

Let $\mathcal{P}(u, v)$ be all $u \sim v$ paths of length at most 3

$$\min \left( \sum_{v \in V} \left( \sum_{e \sim v} x_e \right)^2 \right)^{1/2}$$

s.t. $\sum_{p \in \mathcal{P}(u, v)} y_p = 1 \quad \forall (u, v) \in E$

$x_e \geq \sum_{p \in \mathcal{P}(u, v) : e \in p} y_p \quad \forall (u, v), e \in E$

$x_e, y_p \geq 0 \quad \forall e, p$

- Standard network design LP relaxation, except non-linear objective
  - Easily solved with (e.g.) Ellipsoid
- Use two different rounding algorithms, trade them both off with greedy
Rounding Algorithm 1

Super simple rounding algorithm:

- Add each $e \in E$ to $H_1$ independently with probability $x_e^{3/7}$
Super simple rounding algorithm:

- Add each $e \in E$ to $H_1$ independently with probability $x_e^{3/7}$

Problem: might not result in a spanner.

If add with probability $x_e^{1/3}$, would be a spanner, would exactly be algorithm for $\ell_\infty$ objective from [Chlamtáč-D ’16]
Rounding Algorithm 2 (Simplified)

- For each \( u \in V \), draw \( z_u \in \mathbb{R} [0, 1] \) u.a.r.
- For each \( e \in E \), draw \( z_e \in \mathbb{R} [0, 1] \) u.a.r.
- Add \( e = \{u, v\} \) to \( H_2 \) if at least one of the following conditions holds:
  - \( z_u \leq x_e^{1/4} \) and \( z_v \leq x_e^{1/4} \), or
  - \( z_u \leq x_e^{1/4} \) and \( z_e \leq x_e^{1/4} \), or
  - \( z_v \leq x_e^{1/4} \) and \( z_e \leq x_e^{1/4} \).
Rounding Algorithm 2 (Simplified)

- For each $u \in V$, draw $z_u \in \mathbb{R} [0, 1]$ u.a.r.
- For each $e \in E$, draw $z_e \in \mathbb{R} [0, 1]$ u.a.r.
- Add $e = \{u, v\}$ to $H_2$ if at least one of the following conditions holds:
  - $z_u \leq x_{e}^{1/4}$ and $z_v \leq x_{e}^{1/4}$, or
  - $z_u \leq x_{e}^{1/4}$ and $z_e \leq x_{e}^{1/4}$, or
  - $z_v \leq x_{e}^{1/4}$ and $z_e \leq x_{e}^{1/4}$. 

New aspect: rounds based on randomness at both vertices and edges

- Sampling at edges: [D-Krauthgamer '11, BBMRY '13, Chlamták-D '16]
- Sampling at vertices [D-Krauthgamer '11, D-Zhang '16].
- First algorithm that does both (?)
Correctness: Regularization

Use [Chlamtác-D ’16]:

- Bucket and prune $u \sim v$ paths
- Get that WLOG, LP solution very regular:
- Loses some polylogs
Fix \( \{u, v\} \in E \).

**Lemma:** If \( \max(d_L, d_R) \geq \tilde{\Omega}(y_0^{-2/3}) \), then Rounding Algorithm 1 will include some \( p \in \mathcal{P}(u, v) \) with probability \( \tilde{\Omega}(1) \).

**Lemma:** If \( d_L, d_R \leq \tilde{O}(y_0^{-2/3}) \), then Rounding Algorithm 2 will include some \( p \in \mathcal{P}(u, v) \) with probability \( \tilde{\Omega}(1) \).

So repeat \( \tilde{O}(1) \) times, get high probability bounds. Union bound over all \( \{u, v\} \in E \).
Correctness: Intuition

Modified Algorithm 1: choose each edge $e$ independently w.p. $x_e^{1/3}$ (instead of $x_e^{3/7}$)

- Get path $p = (e_1, e_2, e_3)$ with probability

$$\left(x_{e_1}x_{e_2}x_{e_3}\right)^{1/3} \geq \left(\min(x_{e_1}x_{e_2}x_{e_3})\right)^{1} \geq y_p$$

- So get each path with the “right” probability, so in expectation get at least one path since $\sum_{p \in P(u,v)} y_p = 1$
  - Issue: Paths not disjoint! Concentration?
Correctness: Intuition

Modified Algorithm 1: choose each edge $e$ independently w.p. $x_e^{1/3}$ (instead of $x_e^{3/7}$)

- Get path $p = (e_1, e_2, e_3)$ with probability

$$\left(x_{e_1} x_{e_2} x_{e_3}\right)^{1/3} \geq \left(\min(x_{e_1} x_{e_2} x_{e_3})\right)^{1} \geq y_p$$

- So get each path with the “right” probability, so in expectation get at least one path since $\sum_{p \in \mathcal{P}(u,v)} y_p = 1$
  - Issue: Paths not disjoint! Concentration?

- Intuition of [Chlamtátě-D ’16]: if paths not disjoint, actually doing much better!
  - Get $n(1/n)^{1/3} = n^{2/3}$ left edges, $n^{2/3}$ right edges
  - $n^{4/3}$ ways to complete a path, get each w.p. $1/n^{2/3}$
  - So get about $n^{2/3}$ paths!
Correctness: Intuition

Decrease sampling probability to $x_e^{3/7}$.

- If paths overlap a lot ($\max(d_L, d_R) \geq \tilde{\Omega}(y_0^{-2/3})$), Rounding Alg 1 still works.
- If not, do something else: correlate at nodes!
- Can’t do this for $\ell_\infty$-metric, but (in this case) can do this for $\ell_2$-metric.
Correctness: Intuition

Decrease sampling probability to $x_e^{3/7}$.

- If paths overlap a lot ($\max(d_L, d_R) \geq \tilde{\Omega}(y_0^{-2/3})$), Rounding Alg 1 still works.
- If not, do something else: correlate at nodes!
- Can’t do this for $\ell_\infty$-metric, but (in this case) can do this for $\ell_2$-metric.
  - Having edges bought only by nodes has too much correlation, ends up with large degrees.
  - Need to mix edges paying for themselves (randomness at edges) with being bought by endpoints (randomness at nodes)
  - Argue that if paths “mostly disjoint”, works well in expectation, and can prove concentration.
Conclusion & Open Questions

- For stretch 3, $\ell_2$-norm: analyzed greedy (tight), hardness of approximation, complicated algorithm to beat greedy.

- What about other $p$, other stretch?
  - Some things generalize.
  - Hardness Analysis of greedy should (some really annoying technicalities)
  - Algorithm 2 should generalize to other $p$
  - Some don’t
  - Better than greedy for stretch $> 3$?
  - Even for $p = 2$, $k = 3$, gap between upper bound and hardness. Better algorithms?

- What about $\ell_p$-norm of degree vector for other network design problems?

Thanks!
Conclusion & Open Questions

- For stretch $3$, $\ell_2$-norm: analyzed greedy (tight), hardness of approximation, complicated algorithm to beat greedy.
- What about other $p$, other stretch?
  - Some things generalize.
    - Hardness
    - Analysis of greedy should (some really annoying technicalities)
    - Algorithm 2 should generalize to other $p$
  - Some don't
    - Better than greedy for stretch $>3$?
- Even for $p=2$, $k=3$, gap between upper bound and hardness. Better algorithms?
- What about $\ell_p$-norm of degree vector for other network design problems?
Conclusion & Open Questions

- For stretch 3, $\ell_2$-norm: analyzed greedy (tight), hardness of approximation, complicated algorithm to beat greedy.
- What about other $p$, other stretch?
  - Some things generalize.
    - Hardness
    - Analysis of greedy should (some really annoying technicalities)
    - Algorithm 2 should generalize to other $p$
  - Some don't
    - Better than greedy for stretch $> 3$?
- Even for $p = 2, k = 3$, gap between upper bound and hardness. Better algorithms?
- What about $\ell_p$-norm of degree vector for other network design problems?

Thanks!