APPROX/RANDOM 2019

D Ellis Hershkowitz

September 23, 2019

These are the notes I took while at APPROX/RANDOM 2019. Most of the talks are from APPROX though a few are from RANDOM. These notes were written while trying to keep up with the talks and so are not free from errors. Cheers!

Contents

1 September 20 **2**
- 1.1 Sagar Kale on Small Space Stream Summary for Matroid Center . . . 2
- 1.2 Rajesh Jayaram on Towards Optimal Moment Estimation in Streaming and Distributed Models . . . 3
- 1.3 Harry Lang on Improved Algorithms for Time Decay Streams . . . 4
- 1.4 Chi-Ning Chou on tracking the l2 Norm with Constant Update Time . . . 5
- 1.5 Runzhou Tao on Streaming Hardness of Unique Games . . . 6
- 1.6 Suprovat Ghoshal on Approximation Algorithms for Partially Colorable Graphs . . . 7
- 1.7 Kent Quanrud on Fast and Deterministic Algorithms for Min k-Cut . . . 8
- 1.8 Devvrit on Robust Correlation Clustering . . . 9
- 1.9 Clemens Rösner on On the cost of essentially fair clusterings . . . 10
- 1.10 Constantinos Daskalakis on Reducing AI Bias using Truncated Statistics (plenary session) . . . 11
- 1.11 Ojas Parekh on Almost optimal classical approximation algorithms for a quantum generalization of Max-Cut . . . 12
- 1.12 Reyna Hulett on Single-Elimination Brackets Fail to Approximate Copeland Winner . . . 12
- 1.13 Manuel Fernandez on The Query Complexity of Mastermind with lp Distances . . . 13
- 1.14 Neeraj Kumar on The Maximum Exposure Problem . . . 14

2 September 21 **14**
- 2.1 Euiwoong Lee on Improved Hardness for 3LIN via Linear Label Cover . . . 14
- 2.2 Aleksa Stankovic on Global cardinality constraints make approximating some Max-2-CSPs harder . . . 15
- 2.3 Sai Sandeep on Rainbow coloring hardness via low sensitivity polymorphisms . . . 16
- 2.4 Goran Zuzic Optimal Adaptivity Gaps for Stochastic Multi-Value Probing . . . 17
- 2.5 Alexander Birx on Improved Bounds for Open Online Dial-a-Ride on the Line . . . 18
- 2.6 Leon Ladewig on Improved Online Algorithms for Knapsack and GAP in the Random Order Model . . . 18
- 2.7 Alon Eden on Max-Min Greedy Matching . . . 19
- 2.8 Shuchi Chawla on Online resource allocation, pricing, and prophet inequalities (plenary session) . . . 20
- 2.9 Ray Li on Lifted Multiplicity Codes . . . 22
- 2.10 Nicolas Resch on On List Recovery of High-Rate Tensor Codes . . . 22

3 September 22 **23**
- 3.1 Mike Dinitz on Approximating the Norms of Graph Spanners . . . 23
- 3.2 Arnold Filtser on On Strong Diameter Padded Decompositions . . . 25
- 3.3 Ivan Mikhailin on Collapsing Superstring Conjecture . . . 26

3.4 Anastasios Sidiropoulos on Routing Symmetric Demands in Directed Minor-Free Graphs with Constant Congestion . . . 27
3.5 Alex Wang on Hardy-Muckenhoupt Bounds for Laplacian Eigenvalues . . . 28
3.6 Ben Moseley on Submodular Optimization with Contention Resolution Extensions . . . 29
3.7 Gunjan Kumar on The Complexity of Partial Function Extension for Coverage Functions . . . 30
3.8 Mathieu Mari on Maximizing Covered Area in a Euclidean Plane with Connectivity Constraint . . . 31

1 September 20

1.1 Sagar Kale on Small Space Stream Summary for Matroid Center

k-center

Defined the k-center problem

Could generalize so that centers have colors; only allowed to choose at most 1 center of each color

This is an instance of matroid center (for this talk think of this problem)

Matroid Center

Could represent fairness constraints

Also data summarization

This work studies matroid center in streaming model

k-Center

Formally defined k-center

NP-hard to get a 2 approximation; exists a matching 2 approximation

Streaming k-Center

Points in universe arrive one at a time; want to compute good solutiob using small, e.g. $O(k)$, memory

Alg could be if when a new points arrives and its further than 2τ where τ is guess for OPT then you add it to your set; just use geometric guessing of τ

Matroid Center Problem

Defined matroids

E.g.s:

1. k-Uniform Matroid
2. Partition matroid: at most c_i elements of color i

Problem is to find an independent set on a metric which minimizes that maximum distance of a point to a point in your chosen independent set

Matroid Center in Streaming

Problem is as above but elements arrive in a stream

Gave results; constant approximations for various domains

Space Lower Bound Intuition

If I guarantee you that there is a center in a tiny region but you don't know which point to store so you don't know which colors to remember

Can reduce from CC problem of index; $\Omega(n)$ communication / space

Algorithm Intuition

While an uncovered point; mark all points in its 2τ ball as covered; add points within τ to partition P

Solve matroid intersection on input matroid and partition matroid

Author's key insight is suffices to take an MIS instead of all points within τ

Open Problems

How to improve tha px ratios

Is there an $O(r)$ space algorithm in two passes

1.2 Rajesh Jayaram on Towards Optimal Moment Estimation in Streaming and Distributed Models

Streaming Model

Data streams

Want to model evolution of some large dimensional vector; initially is the 0 vector; get a coordinate-wise update Δ at each time step

Two models studied

1. Insertion only model $\Delta \geq 0$
2. Turnstile: Δ can be whatever

Want to do this with poly log space

Classic problem is moment estimation for varios p

Has been shown that for $p = 2$ can estimate moment with poly log space up to an ϵ

This is tight for turnstile streams

OTOH there's a gap in the insertion model where the lower bound is $\Omega(1/\epsilon^2 + \log n)$ but the best upper bound $O(\frac{1}{\epsilon^2} \log n)$

This work

1. For $p > 1$ rule on large class of techniques
2. $p \geq 2$ improve upper bounds

Message Passing Model

A play at each vertex in a graph; each plaer recieves an inpute; players want to work together to estimate the p-moment of all of their inputs; in each round they can only send messages across their incident edges

Streaming is a special case of message passing; just take the directed line graph; max communication across any edge here is the space used by the streaming algorithm

Study message passing model because nearly all lower bounds in streaming come from this model; in fact come from 1 of 3 topologies

1. 2-party model
2. coordinator model
3. blackboard model

All 3 models have diameter ≤ 2.

This Work

For $p > 1$ in this work they show for any such topology you can do $O(1/\epsilon^2)$ max communication for moment estimation; logarithmic dependence on graph diameter

Second result: for $o \leq 1$ can estimate with max communication $O(1/\epsilon^2)$

Indyks p-Stable Sketch

Generate a kXn matrix where each entry distributed from a "p-stable distribution" where if you look at sum of X_is get back another p-stable distribution

Some facts: they have heavy tails for $p < 2$

Easy solution for F_p estation: fix a spanning tree; merge sketches up the tree

For this sketch needed to send an innter product; hope is that can just estimate this inner product; just round inner product to nearest power of $(1 + \epsilon)$

This doesn't work for blackboard model because of how the errors from rounding compound

Summary

All lower bounds in streaming come from message passing model with small diameter

Show that cannot improve the lower bound for moment estimation on these models

Also improve upper bounds

Still want to close gap for $p \in (1, 2]$

1.3 Harry Lang on Improved Algorithms for Time Decay Streams

Time-decay Streaming Model

Difference from streaming model in that points have decreasing importance as points get older as given by a weight function

Insertion-only is just constant weight function

Sliding windows model is a step function (just consider last n points)

Two weight functions

1. Polynomial decay t^{-s}
2. Exponential decay

This Work

For polynomial decay give a coreset construction

For exponential decay: can solve for a large class of problems

Coreset

What's a coreset? Given a set of ponts can reduce to a subset of points such that for a function of interest can reduce data size such that function's value is essentially the same on the smaller set

Poly Decay Results

Main result for poly decay is get an extra $\epsilon/\log n$ space

General idea is merge and reduce: build coreset of corest of corset etc.

Problem here is weight is different; can't just build a coreset and be done; here there reaches a point in time where can just do that; older things have more negligible differences in weights

The idea here that's different

Exponential Decay results

Problem is ratio is never changing so differences don't become negligible as in poly decay; can't do same thing as before

Do something for k-median clustering

Here also need aspect ratio of set

Use online facility location algorithm

1.4 Chi-Ning Chou on tracking the l2 Norm with Constant Update Time

Steaming Algorithms

Consider insertion only streaming

Want to output some statistics of the input streams: e.g. norm of frequency vector (histrogram of input streams)

Lots of applications

Goal usually is randomized algorithm with sublinear space

In this talk will actually focus on time complexity in streaming model

l2 Estimation

3 types of guarantees can home for

1. one shot estimation
2. weak tracking
3. strong tracking

In this talk interested i weak tracking (usually strong tracking is too strong)

Linear Sketch

Standard class of algoritms is linear sketch

Use randomness to generate sketching matrix Π and sketching vector

Space complexity is small

Update Time

Formally define time complexity as number of field operations needed for each update

Natural question is can you get a faster linear sketch for l2 estimation than standard estimation

Answer is yes; however, only for one-shot estimation guarantee but people also care about estimation for **each** time step (tracking)

This work shoes that can use CountSketch with $O(\epsilon^2)$ rows can achieve weak tracking and a constant time update

Proof Sketch

Idea: exactly one non-zero entry in each column

Thorup et al. showed CountSketch achieves one-shot l2 estimation guarantee; analysis is very simple

Point here is want to show weak tracking guarantee

- Natural attempt: union bound on one-shot analysis
- Instead: use chaining argument to get a fancier and tighter union bound

Step 1: Extracting the Correlation

Intuitively because you're in insertion only model after one time step your strings are very correlated so naieve union bound is bad

Rewrite error terms in some quadratic form

Step 2: ϵ-net

Use a sequence of nets which is finer and finer

What's Missing

- Dudley's inequality
- Bound the size of nets
- Bound the error magnitude (Hansen-Wright inequality)
- High probability regime

Conclusion

First streaming algorithm for weak tracking l2 with constant update time

Future directions: empirical performance of CountSketch

Weak tracking for other norms

1.5 Runzhou Tao on Streaming Hardness of Unique Games

Background

Unique games: a CSP on a graph where assign each vertex a value in $[p]$ and each edge has a constraint on each edge which says what allowable configurations are for endpoints of each edge; want to maximize number of sated assigments or approximatly maximize

By unique games conjecture: hard to approximate

Related to many inapproximability results

This Work

Consider unique games in streaming model: edges (constraints) arrive one by one; li ited space; output a number to estimate the solution

2 trivial algorithms

1. count number of edgs and output $1/p$; $O(\log n)$ space
2. sample $O(n)$ edges and brute force: $O(n)$ space

Prior work

CSPs in streaming model: beating 1/2 approximation for max cut requires $\Omega(n)$ space

A 2/5 approximation for max DICUT

This paper: unique games is hard; i.e. need $\Omega(n)$ space to beat $1/p$ approximation

Method

Reduction from p-ary hidden matching problem: Bob needs to distinguish between random vector and one corresponding to incidence matrix

Show this requires $\Omega(\sqrt{n})$ bits

This problem is closely related to unique games in streaming

Conclusion/Future Work

Need $\Omega(\sqrt{n})$ space for beating $1/p$ apx in unique games streaming

Maybe can use this for lower bounds for other CSPs

Make this a linear bound?

1.6 Suprovat Ghoshal on Approximation Algorithms for Partially Colorable Graphs

k-Coloring

Defined k-coloring

Hard to approximate

Of particular interest is 3-coloring: best known apx is $O(n^{1.996})$-apx

Hardness-wise we know less

NP-hard to approximate using 5-colors

Partial k-Coloring

This talk is about partially k-colorable graphs

Graph is partially k-colorable if there are $epsilon \cdot n$ nodes can delete to get a k-colorable graph

Need to allow an algorithm something more: delete some vertices, color remaining

Not a new problem: e.g. special cases like Odd Cycle Transversal have been studied

Adversarial and Semi-random Models

Graph partitioned into good and bad set

Induced graph on good set is also k-colorable

In adversarial setting adversary can put w/e edges between good and bad set. In semi-random setting add iid edges from good to bad and then adversary adds arbitrary edges

Results

Efficient approximation in adversarial setting; $n^{.25}$ colors

In semi-random model do better; delete $O(\epsilon n)$ vertices and then use $O(n^{1.966})$ colors

Coloring Algorithms

SDP formulation; vector for each vertex; SDP enforces all vectors at least $1/3$ apart; then apply hyperplane rounding; this process gives an approximation based on the largest degree; the fix is to first reduce the max degree in the graph

Challenges

Basi SDP is not necessarily feasible

Rounding SDP is not obvious since SDP doesn't work

Degree reduction rely on combinatorial properties of 3-colorable graphs

A New SDP

Build on previous SDP to fix these problems

Don't require every edge to be satisfied exactly

Have variables to stand for badness of vertices; can at least show this SDP is always feasible

Delete vertices with large w; surviving edges approximately satisfy the coloring constraint

Now in degree reduction step nodes look locally bipartite; apply some Ramsey theorem to locally color these vertices

Finally existing randomized rounding works if inner product constraints are slightly perturbed

Semi-Random Model

Challenge here is to better than in the adversarial setting

Two Cases

1. Many disjoint short odd cycles
2. Few disjoint short odd cycles

When many short odd cycles they'll show up in neighborhoods of bad vertices

On other hand can just delete vertices in latter case

Open Directions

Matching existing LBs

Efficient algorithms for general k

Better inapproximability

1.7 Kent Quanrud on Fast and Deterministic Algorithms for Min k-Cut

Defined min-cut

Minimum k-Cut

Like min cut but have to break into k or more components

For min cut can fix s and try all t and do max flow min cut

Question of can you do better than n times max flow

Answer has gotten all the way down to $\tilde{O}(m)$ for randomized and then even deterministic for unweighted

For k cut max low is not enough

Problem is NP-hard if k is part of the input; can get exact algorithms if k in the exponent though; exact dependence on k is open

Approximations

What about approximations?

1. Algorithm that builds a Gomory-Hu tree
2. Recursive min-cuts

Running time?

Could build a GH tree in $n\cdot$ max flow time. Recursive min cut is \$k $\cdot$ \$ min cut time

Question: is 2 the right approximation? it's ETH hard to do better

A gap for randomized Vs deterministic in k-cut like in the min-cut case

Q1: can you get a deterministic 2-apx for k-cut as fast as the randomized $O(mk)$ algorithm

Q2: can we get 2-apx k-cuts as fast as min-cut?

This Work

$2+\epsilon$ apx in basically $O(m)$ time deterministically

Outline

Solving an LP for $1+\epsilon$ flow

Rounding of the LP

For $k=2$ end up with a pure covering LP; if we look at the dual we have a pure packing LP

Real annoyance was the ≤ 1 constraints on each edge which mess up pureness of packing/covering

Knapsack Covering Constraints

Given a covering integer program w/ multiplicity constraints

Write down the residual LP after maxing out all x_j in some set S

Allows you to drop multiplicity constraints

Apply this to KC LP: force subset of edges to be in solution

Apply MWU framework

1.8 Devvrit on Robust Correlation Clustering

Outline

1. Correlation clustering
2. Robust versions
3. Results / open problems

Correlation Clustering

Told whether points are similar or not; just based on this you have to cluster

Formally, graph where each edge is "similar" or "dissimilar"

Have to output a clustering with minimum cost: pay sum of positive across clusters and negative within clusters

Known results: best is a 2.06 approximation using LP rounding on complete graphs or $O(\log n)$ on general graphs

Robust Clustering

Now also have to detect outliers and not get affeted by them

Same as before but a budget of m vertices that can be removed

Question: is correlation clustering inherently robust

Yes, unlike k-means/k-median

Answer is yes as per this work

Can just do regular correlation clustering optimally and then remove worst m vertices; problem is can't solve CC optimally

So look at this problem in the approximate setting

Exist examples approximate clustering where can't remove a small number of vertices to get down to low cost

Results

1. NP-hard to get any finite approximation for robust correlation clustering
2. For complete graphs have to delete as many as $2m$ vertices to get finite approximations
3. For general graphs need to delete as many as $O(\sqrt{\log \log n}) \cdot m$ vertices to get finite approximation

This motivates a bicriteria setting

Give a (6,6) bicriteria on complete graphs $(O(\log n, O(\log n))$ on general graphs

Central Idea

Bad triangle with 2 similar and 1 dissimilar edges; no matter how you cluster these vertices they add 1 to the cost

For complete graphs:

Step 1: preprocess and remove bad triangles Step 2: apply previous algorithm for LP rounding

For general graphs: flip the approach Step 1: run the clustering algorithm Step 2: run the post processing step

1.9 Clemens Rösner on On the cost of essentially fair clusterings

Clustering problems

Partition points to optimize some objective functions

They consider: 3 k-objectives; k-supplier where centers differ from demands; facility location

Consider **fair** clusters: every point has a color; importantly every points has some color; a subset of points is fair if every color has the same ratio

Introduces a relaxed version of fairness; give an upper bound lower bound for fairness of set

In this work improve exact approximations and gave approximations if the fairness is violated

Some technical results: NP-hard to find the best fair assigment for a given set of centers even for only two colors; Fair clustering problems can be written as integer programs -> LP relaxation can be computed

Algorithm

First compute a good unfair solution; also compute a fractional solution that is fair; then combine two solutions

1.10 Constantinos Daskalakis on Reducing AI Bias using Truncated Statistics (plenary session)

Use statistical approaches to remove biases from training ML models

High-Level Goals

There is selection bias in how training data is collected, i.e. train $\neq$ test

This work tries to decrease the bias coming from this effect using

1. truncated statistics (samples from outside observation window)
2. censored statistics (same but told cound of hidden data)

Caused by

- limitations of measurement devices
- limitations of data collection

Motivating Example: IQ vs income

Question of whether IQ improves income for low skill workers (e.g. from econometrics)

Sample (IQ, training, education, ...) of people with income below poverty line and earnings

Do regression

Obvious issue: fact that you're thresholding incomes may introduce bias

Motivating Example 2: Height vs Basketball

Suppose we just collect data from NBA

Might conclude height is neutral or even negatively correlated (short guy in NBA probably a really good player)

What happened

Only observe ys above some threshold

Motivating EG 3

Truncation on the X-axis

Accuracy of gender classifiers very bad for darker skinned women

One explanation is training data contains more faces of lighter skin tone

Menu

- Motivation
- Flavor of models, techniques, results

Problem 1: Truncation on the Y-Axis

Some subset of data is thrown away according to probability given by function $\phi(y)$

Given filtered data want to recover Θ

Given results based on gradient descent

Give computationally and statistically efficient recovery of true parameters

Compared to literature: alot of work on truncated/censored regression with 2 technical bottlenecks

1. Convergence rates not great: $O_d(1/\sqrt{n})$
2. Computationally inefficient

This work gets optimal rates of $O(\sqrt{d/n})$

Assumes that average x_i has some non-zero probability of not being pruned

1.11 Ojas Parekh on Almost optimal classical approximation algorithms for a quantum generalization of Max-Cut

About a quantum problem but with classical algorithms

Quantum Speedup

Shor's algorithm

Grover's search algorithm

Provable advantages not w.r.t. running time

Quantum Bits Live in a Sphere

qubits can be thought of as unit vectors

Quantum algorithm

Apply a sequence of unitary operations to qubits

An intimate connection between discrete optimization and physics: nature tends towards stable states

Hacking Nature to Solve your Problems

1. map solution values to energy levels
2. realize said physical system
3. let nature relax to a stable low-energy state

Conjectured that problems in BQP outside of polynomial hierarchy

Considering classical algorithms in P for problems in MA; want to get approximations

Polynmials and Quantum Soutions

Can represent CSPs as polynomials

1.12 Reyna Hulett on Single-Elimination Brackets Fail to Approximate Copeland Winner

Sporting Competitions

Defined single-elimination bracket and round robin; in round-robin learn much more

Question is what are we losing if we just play a single-elimination bracket?

Outline

1. Model
2. Related work
3. A surprising answer
4. Conclusion

Model

$n = 2^m$ competitions; deterministic

Can represent a round robin with directed complete graph

Copeland score defined as out-degree in this graph

Question is how well does winner of bracket approximate the best Copeland score winner

Random competitor gives an approximation of $1/2$

Answer depends on how the bracket is seeded

Related Work

"bracket-like" structure achieves a raio of about $2/3$

Any Copeland winner can win a single-elimination bracket if seeded appropriately

Worst Case Seeding

Winner of single elimination must beat at least $\log n$ teams; but could be that they don't beat any other; gives basically a $\log n/n$ approximation

Optimal format with $n - 1$ games has ratio of about $\sqrt{n}$

Random Seeding

Randomly-seeded single-elimination bracket is much worse that $1/2$

Shown by a simple tournament

Conclusion

Single-elimination brackets randomly seeded fail to approximate Copeland winner

Open Questions

Random tournaments instead of random tournament graphs

Tradeoff between approximation ratio and number of games

1.13 Manuel Fernandez on The Query Complexity of Mastermind with lp Distances

Mastermind definition

2 players: codemaker and codebreaker

Codemaker picks colors; codebreaker gets back series of feedback if their colors line up

Mastermind with Lp Distances

Equivalent to hidden vector problem; can consider where have LP distances

1.14 Neeraj Kumar on The Maximum Exposure Problem

Problem Description

Given points in the plane and rectangles that cover them

Want to remove k rectangles to expose as many points as possible

Motivation

Reliability of coverage: adversary wants to choose k facilities to disable coverage for as many clients as possible

Hardness

Geometric version of densest k-subhypergraph problem

Problem surprisingly not easier with rectangles; still hard to approximate within a $O(n^{1/4})$ factor (conditional on dense vs random conjecture)

Can we do better for arbitrary rectangles?

Yes; what this work is about

Basically, skinniness of rectangles is central to hardness

Get bicriteria algorithm with essentially greedy

For translates of a single rectangle: scale so all rectangles are squares; use 2 dynamic programs

Summary

Redefined max exposure

Hard to approximate with restricted rectangular ranges

Has a PTAS for unit-square ranges

Simple bi-critera approximation

Open: Does there exist a constant for arbitrary squares?

2 September 21

2.1 Euiwoong Lee on Improved Hardness for 3LIN via Linear Label Cover

3LIN

Input: system of linear equations over $\mathbb{F}_2$ where each equation has 3 variables

Output: assigment to variables to satisify as many equations as possible

If value is 1 (i.e. can satisfy all); can find satisfying assignment by Gaussian elimination

Value always at least 1/2 by random assigning

Natural question is if value is close to 1 can you do \$.5 + ϵ\$

Gap version of problem where want to know if value is larger than c or less than s

Hastad showed NP-hard to do this for any constant $\epsilon > 0$

Moshkovitz and Raz showed can take ϵ as small as $1/(\log \log n)^c$ for some c; improved to any c

Their result shows ϵ as small as $1/(\log n)^c$ for any constant c

Label Cover

Input: set of constraints, each involving 2 vars

Output: assigment of label to vars

Can define gap label cover as with 3LIN

Parapemters: n = num vars, l = num labels

Hastad reduced from gap-label cover

Compared different values of n and l for previous gap-label reductions

Bottleneck for recent papers is the resulting gap-3LIN ends up with size poly in 2^l; question is how to design a more efficient reduction in l

Linear Label Cover

Variables partitioned into sets X and Y

Input: set of linear constraints $y_j = A \cdot x_i + b$

Output: assigment that satisfies number of satisfied equations

Unlike label cover which is hard this problem is easy when it is perfectly satisfiable by Gaussian elimination

But Gap-LC is NP-hard with same ϵ

By reducing from this they get a reduction with size polynomial in l

Hardness of Gap-LLC

Only remaining thing to do; they show this

2.2 Aleksa Stankovic on Global cardinality constraints make approximating some Max-2-CSPs harder

Max-Cut Problem

Input: graph G

Want to split vertices to maximize the number of edges between the two sets

Approximability: random cut gives .5 of edges; Gomeans-Williamson shows $.867$; NP-hard to get better than a $16/17$; Goemans-Williamson is optimal assuming the UGC

In this work assume the UGC

Max-Cut with Cardinaly Constraints (CC-Max-Cut)

Perscribe sizes of parts to be k and $|V| - k$

Question: how does hardness change with k; e.g. if logarithmic can always solve in poly time; is case where $k = |V| - k$ always the hardest one?

Gave graph showing bounds achievable for different k

This work gives new hardness results; shows that $k = |V|/2$ isn't hardest instance

Max-k-Vertex Cover

Split vertex set into two parts such that number of edges touching one of the parts is maximized; a special case of max-2-SAT

Gives a graph showing approximability and hardness of these problems; show previous algorithms optimal. Again use idea of "adding dummy variables" to flatten hardness curves

Open Questions

Can we match approximability with hardness for every q?

What happens on regular graphs?

What happens for other 2-CSPs?

2.3 Sai Sandeep on Rainbow coloring hardness via low sensitivity polymorphisms

Rainbow Coloring of Hypergraphs

A rainbow coloring of a k-uniform hypergraph: assign one of r colors to vertices so that on every edge all colors are present

NP-hard even for $r = 2$ and $k = 3$ (NAE-3SAT)

Since it's hard want to look at approximate versions

One difference from graph coloring is more colors here makes the problem harder

So an approximate version of the problem is coloring it with fewer colors

Interested in case where r is very close to k; if $r = k$ there is a polynomial-time algorithm

Existing Hardness

NP hard to color with $O(1)$ colors for $k/2$ colors among other results

Show NP hard to 2 color for $k \leq 6$

Approach

View the problem as a **Promise CSP**

Study the **polymorphisms** of the Promise CSP

Show polymoprihsms are **skewed** which implies NP-hardness

CSP

Defined CSPs

In a promise CSP each predicate has a weaker and stronger form; question is can we distingiosh between the stronger and weaker forms (if you satisfy the weaker one then you always satisfy the stronger one); the computational problem is even the stronger form can be satisfied or even the weaker ones cannot be satisfied

Canonical example of promise CSP: differentiate between graph colorable with 3 colors and not colorable with even 6 colors

Dichotomy

Every CSP on a finite set of predicates is either in P or NP complete. Boolean case fully characterized

Much weaker understanding for PCSPs

Polymorphisms

A key tool in PCSPs: way to combine solutions into other solutions; a "discrete convexity"

Odd majority is a polymorphism for 2SAT as is the dictator function (trivial)

No polymorphisms exist for 3SAT

Takeaway

Existence of symmetric polymoprihms -> algorithms

Skewed polymorphisms -> hardness

In this work apply this principle to rainbow coloring problem

Prove that polymorphisms for rainbow coloring are “skewed”

2.4 Goran Zuzic Optimal Adaptivity Gaps for Stochastic Multi-Value Probing

Motivating Example

Suppose want to go to a birthday party. Didn’t really plan for it. Only 1 hour to get all the stuff you want for the party. Want to get a gift, chocolate and a birthday card. Might turn out that store is closed / doesn’t have gift that you want. Model this as a probability.

If all probabilities are 1 and all nodes are distinct this is the “orienteering problem”

The probabilities are independent

The objective is the number of distinct items

Constaint is 1 hour in the given metric

Goal is to maximize the expected objective value

Problem Definition

Given: universe $[n]$; probabilities $p_1, \ldots, p_n$; probing constraints (subsets of universe) which are downward closed; monotone valueation function

Then: find a subset of universe satisfying constraints; then nature samples active elements

Adaptive vs non-adaptive Strategies

Adaptive: a decision tree where e.g. visit store and if it does have an item or not determines where you go next

Non-adaptive: prep-plan your strategy in advance

Natural question is what is the difference between these two types of strategies? I.e. the adaptivity gap

Why care?

Optimal adaptive strategy can be exponentially size and hard to compute / represent

Non-adaptive easy to represent and easier to find; e.g. in submodular case

Small Adaptivity Gap

Allows you to compute good approximations to the best adaptive algorithm by just computing a good non-adaptive strategy

This Work

Show that adaptivity gap is at most 2 if valuation function is monotone submodular

Adaptivity gap is between k and $O(k \log k)$ if the function is weighted rank of k-matroid intersection

2.5 Alexander Birx on Improved Bounds for Open Online Dial-a-Ride on the Line

Elevator Problem

Controlling an elevator. How to react to requests?

Could react immediately but might be inefficient.

Topic of this talk is competitive analysis for elevator problem (dial-a-ride-problem on the line)

Defined problem formally

Goal is to determine best competitive ratio

What is known?

Upper bound of 2.94 for dial-a-ride

Lower bound is 2.04

Their contribution is computetive ratio in $[2.05, 2.67]$

Lower bound: Idea

2 requests where if algorithm behaves in some way get ρ competitive

Next, force algorithm to behave this way

Upper Bound: Idea

1. Serve known requests optimally
2. Ignore new requests until finished
3. Repeat

Problem: reacting immediately is bad

Solution: wait for a certain time

2.6 Leon Ladewig on Improved Online Algorithms for Knapsack and GAP in the Random Order Model

Introduction

Defined knapsack problem

Generalized Assigment Problem: generalizes knapsack; multiple resources of different capacities; an item has different size / value when put into different knapsacks (i.e. resources)

Online variant: items revealed one by one; each item must be packed / rejected immediately on arrival

They consider a relaxed online model where adversary still sets items' costs / sizes but the items are presented in a random permutation

Previous work shows no constant randomized or deterministic algorithms in general but if in random order model can get a constant competitive ratio

Results

Improve competitive ratio for both knapsack and generalized assigment problems; in this talk focus is on knapsack result

Old idea is to split items into large and small item sets where large is larger than 1/2 of the size of the knapsack (so can pack at most 1 large item in the knapsack)

Makes the problem for large items the secretary problem; small items solved with an LP; then combine the two algorithms with a random coin flip

New approach: change large to be $1/3$: becomes 2-secretary algorithm; for small items do greedy algorithm; instead of randomly deciding between strategies, both are performed in a sequential way

Large Items

For large items adapt ideas from 2-secretary problem for the 2-knapsack problem where get to choose at most 2 items for knapsack

Crucial property used is either 1-2 large items are packed with high profit or the knapsack is left empty with sufficiently high probability

Small Items

Use fractional knapsack algorithm; easy to attain a solution for the fractional knapsack problem; can easily solve thi problem; then sample with probability according to the fractional solution

2.7 Alon Eden on Max-Min Greedy Matching

Max-Min Greedy Matching

Given a bipartite graph with a perfect matching

2 players; one is maximizing player, other is minmizing

Maximizer chooses a permutation on the right side; the minimizer then gives a permutation on the left

Vertices arrive according to the left side and match to the highest unmatched vertex on the right side

Gave example where resulting matching is size 2 but there is a PM of size 3

Observations

Resulting matching is maximal and so get at least 1/2

Q: can maximizer beat half

Gave example were can't beat 2/3, namely a cycle on 3 vertices

Motivation

Pricing in markets

A seller sets prices on m items; buyers with different valuation functions. An allocation partitions the goods and the goal is to maximize social welfare; was shown that for general markets can always get 1/2; this work wants to understand if 1/2 can be beat for the simplest kinds of markets

1st Naive Attempt

Give lower degree vertices higher rank

But doesn't work; gave example

2nd Naive Attempt

Try an arbitrary π, see what the worst response is and if the matching is of size about $1/2$ put the unmatched vertices higher in the maximizer's ranking.

Problem is possible to get $1/2n$ for $\log n$ iterations and then a perfect matching

3rd Naive Attempt

Choose a random ordering

Breaks if some vertices are fully connected and rest are matched to exactly 1 vertex

Result

Get \$.51\$ and can compute π in polynomial time

Key concept: Exchange Graph

Compute a perfect matching M, add a directed edge (b, b') on maximizer side if b can steal b' match in M. A cycle along these edges corresponds to a alternative perfect matching

Theorem: if graph has a unique perfect matching then can compute it (above graph is acyclyic): choose permutation according to topological sort

Other Idea: Path Cover

A subgraph of disjoint paths that covers the vertices

In algorithm first compute maximal: to do so start with trivial singleton cover and then repeatedely merge paths; also do path unbalancing where make longest path longer and shortest path shorter; stop when can't merge or unbalance; process ends in polytime

Then use properties of maximal path cover

Open Problems

What if general cost functions

2.8 Shuchi Chawla on Online resource allocation, pricing, and prophet inequalities (plenary session)

Talk to introduce people to problems in online mechanism design

Allocating Limited Resources

Objective: maximize social welfare subject to supply constraints

People have different preferences over items; assign items to people to maximize total social welfare

Algorithmic challenge of give what to who

Also **incentive** challenge: don't know users' values; must trust them to reveal these values; want to incentivize truthful reveals

Celebrated **VCG mechanism** shows if you can solve problem then you can get people to reveal preferences

Some Computationally Simple Settings

Matching setting: each buyer just wants 1 item

Interval packing: items have a total ordering and buyers only assign values to intervals

This Work

Focus of this work: *online setting* in which buyers arrive over time; allocate goods to each buyer as they arrive

Algorithmic challenge: online algorithm competitive against hindsight OPT

Economic challenge: buyers shouldn't misreport values and shouldn't delay arrival

Without further assumptions no solutions

A stochastic-online model: initial input is n distributions; stochastic step in which vales instantiated (even simpler buyers could appear or not appear); then buyers show up in adversarial order

Outline for rest of the talk

- Simple setting; aka prophet inequality
- Pricing as an online mechanism
- Two approaches
  - Dual prices
  - Balanced prices
- Challenges

Simplest Setting

Single item for sale

An online algorithm is just a stopping rule as you iterate over buyers and see what they'll pay

No online algorithm can be better than 2 competitive in this setting: can be seen by first buyer has value 1 and second buyer has $1/\epsilon$ w/ pr ϵ and 0 otherwise

On the upper bound side there is a threshold t such that accepting the first reward more than t is 2-competitive

- This algorithm has a nice economic interpretation: place a price tag on your item for t and just sell it to the first person that offers t

A lot of future work on prophet inequalities in past 10 years

Online resource allocation (the problem today) is a bit different: not just making accept/reject decisions; commit to allocation so a more general space

One thing nice about these works is all the algorithms are threshold rules

Grocery Store Mechanism

Each buyer purchases their favorite bundle while supplies last; assume buyers maximize their value minues their price

Our job is to set prices so that this greedy online algorithm performs well

Might ask why should good prices exist

Prices as dual variables

Gave natural LP for allocation

Taking dual get prices naturally as variables

Complementary slackness gives that if assign set to a buyer then that set must be one of their favorite bundles under the pricing given by the dual

Then question is can we use dual prices in practice

But problems:

1. dual prices are too low because trying to clear the market even though want to keep some items around
2. complementary slackness is not always useful because of stochastic arrivals; solution quickly differs from what LP "intended"

Balanced Prices

Alternative to dual prices; designed to deal with problem 1 above

Not very good when buyers desire bundles

E.g. consider one buyer who pays 1 dollar for any single item but second buyer pays n-1 for n items. Optimal is n-1 but ALG would give 1

Interval Scheduling Setting

Items are totally ordered; buyers desire intervals

2 main results

1. can design competitive mechanisms using balanced prices if bundling allowed
2. can design competitive mechanisms using dual prices if a "large supply" (and some other assumptions)

2.9 Ray Li on Lifted Multiplicity Codes

1 Minute Version of the Talk

A set of strings over a finite field

Disjoint repair problem means every symbol has several disjoint sets that could repair that symbol

Lifted multiplicity are ways to design codes

Outline

Disjoint repair group property

Basic examples

Defined some coding theory jargon

Locality is Useful for Distributed Storage

Locality is useful for distributed storage

Most coding theory about error tolerance

But today we're intereted in locality; roughly about just correcting one error: e.g. how many servers to fix one server that fails

2.10 Nicolas Resch on On List Recovery of High-Rate Tensor Codes

Error-Correcting Codes

Code is a map from messages to codewords or subset of alphabet of length n$$

Other coding definitions

List Decodable Codes

Instead of uniquely determining the codeword we get to output a list of $\leq L$ codewords with guarantee that actual sent message is in the list

Nicely interpolates between Hamming's worst case model and Shannon's random model

Applications in TCS

1. Complexity
2. Cryptography
3. Learning theory

Prior Work

Give result that derandomizes previous near linear-time decoding algorithm

List Recoverable Codes

Receive tuple of lists and want to output all code words s.t. large fraction of code is in some set

Tensor Codes

Given some linear code C and then codewords are tensors of C

This Work

3 new results on list recoverability of tensor codes

First Result

Given high rate base code; tensor code with itself gives a high rate code with deterministic near-linear time; then standard (expander) tricks make the code capacity achieving

Second Result

Combinatorial lower bound on the list size of any high-rate list recoverable tensor code

Summary

Two results on tensor codes

1. Deterministic near-linear time list decoding / recovery
2. Lower bound on list size

Open Problems

1. Truly linear-time capacity-achieving codes?
2. Capacity-achieving linear codes

3 September 22

3.1 Mike Dinitz on Approximating the Norms of Graph Spanners

Message: some definitions and qualitative behavior that we don't fully understand

Spanners: basics

A subgraph that preserves stretch up to t

Sufficient to hold for each edge

Classical Objectives

Want small stretch and small "cost"

Two natural costs: total edges, maximum degree

Theorem 0 of graph spanners: all graphs have a $2k-1$ spanner with $O(n^{1+1/k})$; this is tight as per the Erdos Girth Conjecture

No such theorem for max degree

Optimizing Spanners

Switch our point of view from tradeoffs to optimization

Previous Work

Main thing to note is that for basic this tradeoff gives you automatic $n^{1.k}$ aproximation; can be beaten for special cases but seems hard to beat that

Motivations and Issues

Number of edges

- pros: natural; nice tradeoff
- cons: huge degree might be really bad

Max degree

- pros: encourages low loads in distributed
- cons: if some node has large degree maybe shouldn't want all other nodes to have large degree

So go to p norm for degrees; interpolates between these two objectives

Introduced this earlier this year in an ICALP paper; proved the equivalent theorem 0 in the p norm setting; was tight

Solved the tradeoff question; now can ask about the optimization question

Results

Greedy is an $O(n^{3/7})$ approximation for l2 3-spanner

$O(n^{5/13})$-approximation for l2 3 spanner

Label cover hardness

Why Greedy?

Why study greedy if a better algorithm?

- Natural and important
- Demonstrates that greedy behaves differently under 2 norm for stretch 3 than under 1 and ∞ norm

For stretch 3:

- l1: greedy a $\sqrt{n}$-approximation
- l infinity: greedy has max degree at most Δ and OPT at least $\Delta^{1/3}$ so an $n^{2/3}$-apx
- Difference is in l2 greedy is an $n^{3/7}$ even though using global lower bounds gies greedy should be a $\sqrt{n}$ approximation

Convex Relaxation

Gave convex relaxation of spanners problems

Rounding Algorithm 1

Independent randomized rounding but where do lp value to the $3/7$

Problem: might not result in a spanner (if did to the $1/3$ would be a spanner)

Rounding Algorithm 2

Correlate at the nodes to ensure feasibility

Previous algorithms did this trick at the nodes for the l1 norm; also did it at the edges; this is the first algorithm that does both

Open Questions

Network design problems that optimize for l_p norm; Laci suggested spanning trees

3.2 Arnold Filtser on On Strong Diameter Padded Decompositions

Clustering Problems: Stochastic decompositions

Desired properties for clustering

- small diameter
- connectivity
- nearby clusters go to same cluster

Definition of padded decomposition (β, Δ)-PD if

1. every cluster has diameter bounded by Δ
2. for every small ball this ball lies in the cluster with some good probability decaying as $e^{-\beta}$

Strong vs Weak Diameter

Weak diameter: maximum distance of vertices in same cluster w.r.t. original metric

Strong diameter: max pairwise distance in induced graph

History and Results

Can get padding $\beta = O(\log n)$ for every Δ

Can get weak diameter with padding of doubling dimension

For K_r minor free can get padding r^3; improved to r^2

For strong diameter can get $\exp(r)$ and even $O(r^2)$

Two questions:

1. Get padded decompositions for doubling dimension with strong diameter
2. Can get strong with $O(r)$ for minor free graphs (focus of talk)

Applications

Many…

Key Lemma

Sparse net gives padded decomposition

Suppose a net $N \subseteq V$ of centers s.t.

1. Covering: every point with Δ of a net point
2. Packing: number of centers bounded by τ within a 3Δ ball of any vertex

Then can get an $(O(\ln \tau), 4\Delta))$ padded decomposition

Proof uses MPX algorithm; explained how MPX works; can think of MPX as vertex just joining cluster to maximize function

For their algorithm choose this shift time cleverly

MPX to Technical Lemma

Each vertex goes to a cluster within 3Δ because using truncated exponential distribution; gave analysis

But minor free graphs don't have any sparse nets; idea is "core clustering"

Core Clustering[AGGNT14]

Grow ball arond vertex, then take new vertex with path to clusters and grow ball around path; iterate

If graph is K_r minor free then each core tree has at most $r$leaves

Lemma: core clusternig is a padding in a sense but diameter might be very large

Cam get sparse net by taking vertices along path

So given a core cluster can create a strong padded decomposition

Thus, create core clustering and then partition clusters into additional clusters using the sparse nets these clusters contain

Open Questions

The family of K_r minor free graphs admits $O(\log r)$ strong padded decompositions

Question is even open for treewidth

3.3 Ivan Mikhailin on Collapsing Superstring Conjecture

Superstring Problem: Overview

Karp original problem

Given a set of strings; want shortest string that contains all of them as substrings

Many approximation algorithms: best is a 2.479 from Mucha 2013

Greedy algorithm is conjected to give a 2

Greedy algorithm

While there is more than one string take two strings with max overlap and replace them with their superstring

Gave tight example for greedy

Failed to show but found a new approach

Hierarchical Graphs

Way to represent a collection of strings: assign nodes to strings: connect longest prefix to string and string to longest suffix

Any path spells a string; going up adds a symbol to string

Can reformulate superstring problem as a graph problem

Given hierarchal graph what is shortest closed walk that goes through all the top nodes

Nice property of graph: collapsing a pair of arcs; if a path through layer above, have path through layer below

Collapsing Algorithm

Take any solution; double it; collapse it as above

Conjecture: the result of this process is the same for all initial solutions

If the conjectureholds then the collapsing algorithm is 2-approximate

An example of how two solutions become 1

Greedy Hierarchical Algorithm

Construct an Eulerian set of edges that passes through all gray nodes and is as small as possible

Gave an example

Weak conjecture: hierarchal greedy is 2-approximate

Strong conjecture: result of hierarchical greedy algorithm coincides with result of collapsing algorithm

Support for Conjectures

True if input strings have length at most 3

Verified on millions of datasets

A framework online to test examples: comsciclu.ru/scs

3.4 Anastasios Sidiropoulos on Routing Symmetric Demands in Directed Minor-Free Graphs with Constant Congestion

Routing in undirected graphs

Edge/node-disjoint paths problem

Given pairs of nodes which correspond to unit demands; want to route so that disjoint

Goal is to maximize number of routed demand pairs

Prior Work

NDP is NP-hard

NDP is FPT

$O(\sqrt{n})$ approximation for EDP / NDP

Slightly better approximation for planar graphs or grid graphs

$\Omega(2^{\sqrt{\log n}})$ in approximability

So hard to approximate so people study (bicriteria) relaxations

Allow youself $O(1)$ congestion; exist poly $\log n$ approximations if you allow this (2 congestion for EDP and $O(1)$ for NDP)

Routing in Directed Graphs

Problem seems much harder in digraphs; for constant congestion c there is an $n^{\Omega(1/c)}$ hardness of approximation

Case of **symmetric demands** seems much more tractable: if want to send from s_i to t_i then also want to send from t_i to s_i; both EDP and NDP versions

A poly $\log n$ approxmiation for the all or nothing flow variant and with constant congestion on NDP on planar graphs

This work: extend these results to arbitrary minor-free grpahs: poly $\log n$ approximation with constant congestion

Well-Linkedeness

α-well-linked if any equal sized subsets of vertices can route a matching from one to other with congestion at most $1/\alpha$

Overview of Algorithm

Same as Chekuri algorithm:

1. Reduce instance to where terminals T are $\Omega(1)$-well-linked
2. Find a large **crossbar** connected to a large fraction of terminals
3. Route a large fraction of terminals through crossbar

Have to modify steps 2 and 3 for this work

Crossbar is some routing structure where have many cycles with same orientation: need to find a large flat grid minor to find these structures; use Roberstson-Seymour structure theorem (apices, vortices, etc.) for minor-free graphs to find this structure

Conclusions

Extended Chekuri result to all minor free graphs

Other tractable instance on directed graphs?

3.5 Alex Wang on Hardy-Muckenhoupt Bounds for Laplacian Eigenvalues

Outline

1. Introduction
2. Bounding λ_2 in terms of graph structure
3. What is Ψ_2?
4. Summary

Graphs and Laplacians

Given vertex masses μ and edge weights κ

Laplacian is $L := D - A$; this depends on vertex and edge weights

Understanding the Laplacian

As a linear map: value at a vertex is the difference of x values weighted by edge weights

As a quadratic form: difference of neighbors where x is now squared

Then L is PSD

Can write down generalized eigenvalue equation $Lx = \lambda Mx$

λ_2 is what we want to bound today (Neumann eigenvalue)

In case where μ is the degrees then λ_2 controls mixing rates of random walks

Why bound λ_2 if I can compute it? Our goal is to give good bounds on λ_2 in terms of graph structure

Cuts and Cheeger's Inequality

Can bound λ_2 in terms of sparsest cut of the graph; can bound below by sparsest cut squared and above by twice it

This work: defines Ψ_2 which is like sparsest cut but don't minimize over partitions; namely, can drop vertices

Can think of Ψ_2 as a relaxation of sparsest cut

This work shows $\Psi_2/4 \leq \lambda_2 \leq \Psi_2$

Gave example comparing their result to Cheeger's; showed their result is better by a factor of n

Graphs with Two Vertices

Can compute λ_2 and Ψ_2; they're equal here

Given path vertex Ψ_2 is sort of like squinting until you just get two vertices: Ψ_2 is sort of the best two-vertex approximation for G

3.6 Ben Moseley on Submodular Optimization with Contention Resolution Extensions

Submodular Maximization

Defined problem

Two classes of considered functions: monotone and **non-monotone**

Usually constraints on feasible solutions are downward closed families: e.g. **interval constraints** (focus of this talk)

In interval constraints each node has a start and end time and a feasible solution is one where chosen intervals don't intersect

Gave prior work:

1. Continous framework: extend submodular function to be continous then optimize and round
2. Local search

Results

Define contention resolution extension

A framework for designing submodular function maximization algorithms under constraints

Use to give a .188-approximation for interval constraints

Algorithmic Tool: multilinear extension

A continous extension: can be thought of as $\mathbb{E}[f(x)]$: i.e. f evaluated on set constructed randomly by sampling each i with probability x_i

Continous Greedy

To optimize miltilinear extension

Intuitively move in direction to improve function

Gets within a constant factor of optimal solution

Traditional Contention Resolution

Normally would just randomly draw elements according to feasible solution; but might end up infeasible; to make this set feasible use **contention resolution** scheme

Improving the Framework

Can either

1. Construct better fractional solution
2. Improve contention resolution scheme

In this work: want to aggregate these two losses into a single step; continous extension should take feasibility into account

New Algorithmic Tool

"Contention Resolution Extensions"

Parameterize by a contention resolution scheme

Sample at each step to see if contention resolution scheme fails and then remove points to make feasibile if it fails

3.7 Gunjan Kumar on The Complexity of Partial Function Extension for Coverage Functions

Coverage Functions

Some universe U with many elements

Some function f which is cardinality of sets

In general a **coverage function** is some fnction on power set s.t. f is the cardinality of the union of the sets

Naturally arise in

- Auctions and other areas in game theory
- ML
- Social network
- Facility location

Partial Function

Not defined on all powerset, just some subset of power set

The extension is s.t. result gives at least what coverage function gives on superset

Problem they're considering is whether exists an extension of a coverage function

If this problem is NP-hard then proper PAC learning not possible (Valiant)

Problem Statement

Stated formally

Results

First result: Coverage extension is NP-complete; reduction from fractional colouring of graphs

Corollary: proper PAC learning not possible;

First such lower bound of learning of coverage functions

Can extend to approximate versions where given function agrees up to a multiplicative α

For approximate extension give d approximation algorithm where d is the max size; d in worst case can be m

For norm extension problem show anything sublinear in sum of evaluations is not possible even if the max size of a set is 2

3.8 Mathieu Mari on Maximizing Covered Area in a Euclidean Plane with Connectivity Constraint

Look at classic problems like dominating set but with connectivity constraints

Connected Unit-Disk k-Coverage Problem

Have unit disks in Euclidean plane

Want to buy k disks so that union is connected and to maximize the area covered by the union of disks

Generalizations

Constant approximations known; can generalize to k-coverage versions where $\Omega(1/\sqrt{k})$-approximation possible; without connectivity can get $1 - 1/e$

A PTAS for unit-disk k-coverage; so want to see what's possible with connectivity

Results

- 1/2-approximation
- PTAS with resource augmentation

Lower bounds

- NP hardness
- APX hardness

Approximation Algorithm

Adapt greedy for k-coverage; add connetivity

At each step of algorithm. add a disk that maintains connectivity

Problem is can end up with an $\Omega(k)$; gave example

Instead could try to add 2 disks at once; this is the 1/2 approximation

Proof Sketch

First phase: S is not a dominating set

Second phase: connectivity is guaranteed; apply classic analysis for maximizing monotone submoular function

The $1/2$ is right

Improving $1/2$?

Taking more than $t = 2$ disks doesn't help; gave example

Can beat $1/2$ and get PTAS if "allowed to cheat" a little bit; namely set is almost connected meaning if can add ϵk disks wherever you want then can get a connected solution

Proof uses shifted quadtree where show a near optimal solution that only crosses at "portals" (Arora's TSP stuff)