9th Workshop on Flexible Network Design

D Ellis Hershkowitz

May 22-25, 2018

These are the notes I took while at the 9th Workshop on Flexible Network Design. I didn't attend every talk and I've omitted the talks for which my notes were not great. The full schedule is available at this URL: `http://www.cs.umd.edu/sites/default/files/Documents/fnd_2018_schedule_flyer.pdf`. These notes were written while trying to keep up with the talks and so are not free from errors. Hopefully these will be helpful to somebody trying to get a general sense of these talks. Cheers!

Contents

1 May 22, 2018

1.1 Aravind Srinivasan on A Lottery Model for Center-Type Problems

k-Center

Gave definition

$\Theta(2)$-approximable

Matroid center

A matroid generalization of k-kenter; set of centers is an independent set in the matroid

$\Theta(3)$-approximable

What about if you allow outliers?

Introduced by Charikar

Only need to cover $\geq t$ clients. Can ignore an arbitrary subset of at most $n - t$ clients.

In this work: how can we incorporate "fairness" in this problem if you repeatedly solve an outliers problem. Maybe "unfair" if same points are always outliers.

The lottery model

Every client submits a probability p_j with which it should be close to a center subject to

- Feasibility constraint: centers are always feasible
- Coverage constraint: With prob 1 at least t clients covered
- Fairness constraint: probability j covered is at least p_j

Gave some results for k-center and matroid-center

Proof idea

Uses standard LP formulation for center problems (along with matroid rank constraints)

Do filtering once you solve the LP: cluster for client j is all i with positive X_{ij} (assignment)

Use Tardos filtering idea for k-median (used this with Ravi before for two-stage problems); first aggregate clients with overlapping balls.

In matroid case have to be careful that clients that open as facilities are actually an independent set.

Algorithm uses integrality of intersection of 2 matroids.

Modify LP by adding fairness constraints.

Matroid trick from Ravi, Singh et al.

Classic Edmond's trick for matroid polytopes The tight constraints for a matroid polytope form a chain (see `http://www.contrib.andrew.cmu.edu/~ra...` page 70).

Wondered if this is a reasonable formulation of "fairness"?

1.2 Debmalya Panigrahi on Random Sampling for Hypergraph and Hedge Connectivity

Edge connectivity

Min number of edges which cutting disconnects the graph (the min cut)

Min cut is important b/c perhaps it represents the most vulnerable cut in a graph

But really networks don't work like that; typically edge failures are highly correlated

Hedge connectivity

Partition edges into color classes. A whole color fails together. Call these color classes *hedges* Now can ask about the hedge connectivity of the graph (how many hedges to cut to disconnect graph)

Seems restrictive but can reprsent any sort of failure (replace an edge by a rainbow path)

Given these color classes, how do I find the hedge connectivity of a graph?

A minimum hedge cut can partition vertices into many sets (i.e. not necessarily a bipartition)

Special case that this captures

Edge connectivity (every edge is its own color)

Hyper-edge connectivity (draw a spanning tree on a hyper-edge; it gets its own color)

Hedge cut function is not necessarily submodular

So is it even poly-time solvable?

Focus on $s - t$ min hedge cuts

This problem is NP-hard and hard to approximate

This paper

Hardness goes away when looking at global hedge-connectivity; i.e. there exists a PTAS. Also a quasi-polynomial algorithm.

Random contraction in graphs (Karger '92)

A simple min-cut algorithm: randomly contract edges until there are only two vertices remaining; output edges spanning the two remaining vertices (BH did this in his class: `http://www.cs.cmu.edu/~haeupler/15859F15/docs/lecture01.pdf`)

Proof: probability of success of a single contraction is like $\geq 1 - \frac{2}{n}$. Then induct.

A natural extension to hypergraphs

Randomly contract hyperedges. Stop once every hyperedge is spanning

Proof: Again know degree of every vertex is at least the min cut value. But in a graph every edge has degree 2 but not true of hypergraph. Best you can do is replace this quantity with avg rank of vertices.

Try to extend on hedges

Algorithm isn't even well-defined. What does it mean for all hedges to be spanning? Even more of the analysis fails.

Suppose every hedge has small rank (i.e. number of incident vertices). This graph must have a lot of edges.

Two nice properties hold:

1. At least half probability won't hit something in a min cut.
2. Still making large progress when contracting outside of min cut.

But eventually the rank of hedges gets close to the algorithm. But not a nice solution. Two possibilities:

1. Every hedge belongs to min cut
2. At least 1 outside of min cut

Algorithm: Choose one of the options randomly in this case

What happens if hedges fail probabilistically?

Sharp threshold of $\tilde{O}(1/k)$ above which a graph is connected if sample each edge with this probability and below its disconnected where k is the size of the min cut.

In graphs all cuts concentrated around expectation but not true for hedge cuts.

New theory of *cut representatives*: although many near-minimum cuts they are built around some core subset of hedges (called hedge representatives).

Main open question: is there a polynomial PTAS for this problem?

1.3 Jaroslaw Byrka on Constant-Factor Approximation for Ordered k-Median

Two sets of points: clients + facilities

Problem Specific constraints (k and capacities)

Objective over clustering (median, means, center, **ordered**)

In ordered k-median only pay for the top l most expensive clients

(Generalizes both k-median and k-center)

Apparently it was already studied a lot in OR research. Not too many algorithms known though.

Known results

k-center: $\Theta(2)$-approximable or $\Theta(3)$ if clients $\neq$ facilities

k-median: gap between lower and upper bound. Integrality gap between 2 and 3

k-means: about $\Theta(9)$ approximable (though a slight gap)

Some previous results on ordered k-median.

Charikar and Li LP rounding algorithm for k-median. Consider a ball around a client capturing a certain amount of facility opening and measure its radius. Cannot open more facilities than the fractional solution does. What Charikar and Li do is form clusters with y money of at least $\frac{1}{2}$. They pair clusters with at least $\frac{1}{2}$ money. Distance between balls with at least $\frac{1}{2}$ money somehow relates to how much of the y money is missing in a ball. Then maybe need to take multiple hops to find your facility. Adapt an unoptimized version of Charikar and Li to their needs.

Problems to overcome: $\mathbb{E}[\max]$ not bounded in $\max \mathbb{E}$.

For the rectangle-weight case

Introduce guessing like in k-center. Focus on rectanglular case where only focus on some clients. Guess T (in the optimum solution the distance of the last client that counts). Then only distance that you have to pay for is distances that exceed T. Under this new distance you basically have a metric up to an additive T.

Open questions:

- Generalize to non-monotone weight function
- Generalize to when clients have speeds

1.4 Mike Dinitz on Optimal Vertex Fault Tolerant Spanners (for fixed stretch)

Definition of spanners: A sparse subgraph that preserves up to stretch t

Also edges are unweighted for this talk.

$t = 3$ for this talk.

Main question

How many edges do I need? Message is that spanners are solved. [Althofer '93] For any graph and any integer k can give a $(2k-1)$ with $O(n^{1+1/k})$ edges. Under Erdos girth conjecture this is true.

Greedy algorithm

just go through edges and add them if you need them (i.e. if they are stretched too much currently)

Correctness is immediate

It's pretty sparse: because there are no 4-cycles in H and a graph with 4-cycles can only have $O(n^{3/2})$ edges. Also there are graphs on $\Omega(n^{3/2})$ edges with no 4 cycles.

The greedy algorithm, then, builds an existentially optimal spanner

In practice

Spanners useful in distributed systems. In distributed computing things fail. If build a 3 spanner and a vertex dies then you're really screwed. In graph still have a short path but in spanner distance is infinite.

Definition: an f-vertex fault tolerant (VFT) $(2k-1) = t$-spanner if for every fault set F on f vertices $H \setminus F$ is a spanner of $G \setminus F$. Key points is you don't have to be a spanner of G without faults.

Questions: how much extra above $n^{1+1/k}$ do we need to pay for f-vertex fault tolerance?

General approach

Build lots of slightly-different spanners and union them.

You'll need basically f spanners if you want to do things in this way

Message of this paper is you can get below f. In particular can get a $\sqrt{f}$ dependency in your spanner density. A variation of the greedy algorithm. And a matching lower bound.

Theorem: All n-node graphs have f-VFT 3-spanners on $O(n^{3/2} \cdot \sqrt{f})$ edges (and this is optimal under the Girth Conjecture)

Algorithm: For each edge add it if there exists a set of faults that cause it to have stretch more than 3 (finding such a set is an instance of shortest paths interdiction or something)

Proof: WLOG can assume graph is D-regular When add an edge can only create $\leq f \cdot D$ 4-cycles

Number of 4-cycles that include x is D^2. So number of 4-cycles including x is about $\frac{D^4}{n}$

So total number of 4-cycles is at least D^4

Putting these two observations together gives number of edges is at most $n^{3/2} \cdot \sqrt{f}$

Generalizing to Larger Stretches

So far only for $t = 3$.

Get sublinear in f (though bad depedendence on k)

Proof for large k not gone into but why doesn't the previous analysis work: get like new edge creates at most fD^3 cycles. This is the wrong bound. Need $\leq (fD)^k$ Intuitively this picture can't happen very often (a lot of 4 cycles in the picture but also not closing out 4 cycles)

Lower Bound

Uses Girth Conjecture.

(Didn't totally catch this)

Open Questions

Edge fault tolerance. Upper bounds work fine for this. But lower bound runs into some issues.

1.5 Mohammad Hajiaghayi on Online Decision-making and Auctions: Prophets and Secretaries

Optimal stopping theory. A sequence of events. At some time want to make an action.

Secretary problem and prophet inequality

Formal definition given.

Secretary

Given random permutation of numbers

A simple tight algorithm: observe the first n/e elements. Once you see an object at least as good as the best of the first n/e then take it. Asymptotic success probability of $1/e$.

Prophet Inequalities

Given known probability distributions $D_1, \ldots, D_n$.

Want to choose a v_τ to maximize the ratio of the expectation $\mathbb{E}[v_\tau]/\,\mathbb{E}[\max v_i]$. Here also a simple algorithm. Compute a virtual price $p = \frac{1}{2}\,\mathbb{E}[\max v_i]$. Select the first item i where $v_i \geq p$. Proof is not hard but cute.

Prophet Secretary

Similar to prophet inequality but boxes arrive in a random permutation

For i.i.d. instances Hill and Kurtz prove a theoretical bound of $1 - \frac{1}{e}$

For non-i.i.d. instances: $1 - \frac{1}{e}$ approximation

Very recent improvement to $1 - \frac{1}{e} - \frac{1}{400}$

They give a .74 apx for i.i.d. Simple tight algorithm: put $F^{-1}(1 - 1/e)$ as the threshold where F is the CDF of joint distribution of the max Can generalize these results to matroid and combinatorial auction.

(Missed part of the talk)

k-server problem

Given a graph. Have k servers on vertices. An online sequence of calls. Have to send servers to demands. Want to minimize total movement (weighted by edge costs). A $(2k - 1)$ approximation by Papadimitriou.

Worst-case model not suitable for applications. In prophet setting we have a distribution D_i for each demand d_i.

Reliable Network Design Problem

Graph G and a vertex root. A connectivity requirement k. Online sequence of connectivity requests. Need to connect to root.

In prophet setting there is a distribution for each request.

Open Questions

We don't know for prophet secretary if i.i.d. is harder or not than non-i.i.d.

1.6 Kamesh Munagala on Fair Allocation of Indivisible Public Goods

About "public goods"; goods that benefit many people at once N voters with M candidates and each voter has a utility for each candidate

Slightly more general setting: participatory budgeting (Knapsack)

Voters have utility for projects; each project costs something; a hard budget constraints (maybe can fractionally fund a project)

What is fairness?

Define fairness via stability w.r.t. *coalitional deviations*

Deviation of subset S of voters: S goes away and constructs their solution with their share of the resources (i.e. $|S|/N$). If they do strictly better then they are called a **blocking coalition**

An allocation is "fair" if there is no blocking coalition. Fair allocations are in the **core**

Properties of core utility U^*.

If S is all voters it encodes your aggregation "is Pareto-optimal"

If S is a singleton voter it encodes "proportionality"

(Didn't catch rest of talk)

1.7 Barna Saha on Fully Dynamic Set Cover-Improved and Simple

Set cover but elements arrive and depart and must maintain a good solution while keeping the update time low (could always recompute from scratch but that would take a long time)

In offline setting two approximations

- $O(\log n)$ approximation using greedy or LP rounding
- f-approximation (f is the max number of sets an element is in) using PD

Unless P=NP not possible to have a poly-time solution with $o(f)$-approximation

For $O(f)$ apx there is a very simple offline algorithm. Pick an uncovered element (pivot element); take all the sets that contain it. This gives an $O(f)$ approximation. An $O(f)$ apx because the optimum has to choose at least one the f you choose each time.

Prior work on dynamic

An $O(\log n)$ approximation with update in time $O(f \log n)$. Also $O(f^3)$ approximation with $O(f^2)$ update time [Gupta STOC 2017].

So in dynamic setting don't even have an $O(f)$ approximation (even though it's trivial when done offline)

Missing: an f-approximation for set cover with low update time

They get an $f(1+\epsilon)$-approximation in $O(f \log n)$ update time

Trivial algorithm works if you only have insertions; but if you have deletions you have to consider how to remove sets from your solution. In particular what do you do if pivot elements are deleted?

Suppose deletions are not arbitrary but random.

Then only do an update if enough pivot elements are deleted

Pick a pivot from the uncovered elements uniformly at random does not quite work for general deletions. But with a slight modification can get it to work

If more than ϵ fraction of elements are deleted have to make some updates. Trick is to only update a suffix (the last few in the stream; because the pivots that are later have higher probability of being a pivot) instead of updating the entire sequence

2 May 23, 2018

2.1 Deeparnab Chakrabarty on Generalized Center Problems with Outliers

Problem generalizes k-center

Minimize client-facility distance

An approximation of 3 is tight if clients and facilities aren't the same set

2 Generalizations

1. Could also add weights to facilities and minimize max distance subject to weight constraints
2. Could also do k-center add with $n-m$ outliers; i.e. only pay for top m most expensive; also has a 3-approximation. Based on combinatorial greedy algorithm

What if we mix the two generalizations?

A bi-criteria 3-approximation by Li IPCO 2013. Left open if a non-bi-criteria 3-apx

Robust $\mathcal{F}$-Center Problem

$\mathcal{F}$ is a down closed familiy of subsets of F

$\mathcal{F}$-center: Find $S \in \mathcal{F}$ to minimize max distance

Robust version: In robust version get $n-m$ outliers too

Complexity of $\mathcal{F}$

Set system $(F, \mathcal{F})$

A subpartition $\mathcal{P}$ of F

$\mathcal{F}$-PCF problem: decide if exists $S \in \mathcal{F}$ such that $S \cap P = 1$ for all $P \in \mathcal{P}$

$\mathcal{F}$-PCM problem: the maximization of the above e.g. colorful knapsack program

Informal Theorem

Given $\mathcal{F}$ if you can solve $\mathcal{F}$-PCM in polynomial time then you have a 3-approximation for the robust $\mathcal{F}$-center problem

Also if you don't have outliers then a solution to $\mathcal{F}$-PCF problem gives you a 3-approximation

An α-apx for $\mathcal{F}$-PCM still gives a 3-apx but then you violate constraints (don't cover all your clients) up to an α factor

Algorithm Ideas

Reducing $\mathcal{F}$-center to $\mathcal{F}$-PCF: pick an uncovered client; draw a radius of $2\,\mathrm{OPT}$ around client; every facility within OPT forms part of a partition; repeat

Observation 1: optimal picks at least one facility from each part -¿ it follows that the $\mathcal{F}$-PCF instances is feasible. Picking a facility in each part of the partition gives a 3-apx.

Now consider doing the same thing for $\mathcal{F}$-center with $\mathcal{F}$-PCM. For each facility in a part, give it a value that corresponds to the number of freshly covered clients. Now this is an $\mathcal{F}$-PCM instance. Observation 2: a solution to the $\mathcal{F}$-PCM instance of total value at least m gives a 3-approximation. But the $\mathcal{F}$-PCM instance must have optimum value of at least m. But now can't argue that every client must be covered since they could be an outliers. Arbitrary ordering of clients hurts.

So now ordering clients via the coverage polytope (an LP). Have a variable that is 1 or 0 if client is covered. Look at all candidate solutions; take a convex combination of these.

Now the algorithm is as follows: (didn't get)

Main theorem: if cov(v)s from the LP are in the coverage polytope then the $\mathcal{F}$-PCM has feasible solution

Proof of main theorem: (didn't get)

Conclusion

Main open question: is there an instance where the optimization problem is harder than the feasibility problem?

2.2 Karthik Chandrasekaran on Hypergraph k-cut in Randomized Polynom1ial Time

Definition of hypergraph. A k-cut set is a subset of edges whose removal leads to k connected components

Alternatively a partition interpretation

In hypergraph k-cut problem want to find a minimum number of hyperedges to remove and get k connected components

Challenges

For each set of k distinct terminal nodes, find min cut between these terminals. But this is multiway cut problem and is NP hard for $k \geq 3$.

But for graphs can solve this in time $n^{\Theta(k^2)}$. Also a randomized algorithm. The gap has steadily been narrowed between these two.

Complexity of hypergraph k-cut is still open.

Previous Work

Certain restricted cases are solvable in poly-time

1. $k = 2$ (hypergraph global min-cut): sufficient to solve hypergraph min s-t cut problem.
2. $k = 3$: also solvable in deterministic polynomial time [Xiao '08].

Constant rank hypergraphs (rank of a hypergraph is the size of the largest hyperedge; e.g. rank 2 is a graph): solvable in deterministic time [Fukunaga '10]

But what about large rank?

This Paper

A randomized polynomial time algorithm to solve hypergraph k-cut for every constant k in arbitrary rank hypergraphs

The Algorithm

For hypergraph global min-cut (this generalizes to k-cut): based on randomized contractions

Hyperedge contraction definition: what you would expect Natural attempt to solve this problem is to run Karger's uniform random contraction algorithm; but doesn't work e.g. one hyperedge of rank n graph.

Observation: don't contract a hyperedge crossing all 2-partitions (happens only if $|e| = n$)

So need to refine Karger's algorithm

Approach: dampen the probability of picking large hyperedges. Could pick hyperedges with probability inversely proportional to ran. They using dampening factor α_e as the $\Pr(v \notin e) = \frac{n-|e|}{n}$ for a uniformly random v

Notice that this is efficiently computatable

Algorithm:

First check if all dampening factors are 0 then return E

Otherwise draw an e proportional to α_e

Contract e and recompute α_{e}s

(If ever have fewer than 4 vertices then just brute force)

Analysis

Polytime since dampening factors are efficiently computable

If input hypergraph is a graph then all dampening factors are the same so it's just Karger's algorithm (uniform sampling)

$q_n :=$ the probability that an algorithm returns a particular min-cut set

Show $q_n \geq \frac{1}{\binom{n}{2}}$ by induction on n

More analysis shows the probability that the algorithm returns a particular min-cut set is at least $1/\binom{n}{2}$

Generalizing algorithm to hypergraph k-cut. Just change dampening factor.

Algorithm also extends to hedgegraph k-cut.

Conclusion

Randomized poly-time algorithm for k-cut problem.

Directions:

- Understand complexity of hedgegraph global min cut problem: NP-hard or poly-time?
- Complexity of hypergraph k-partitioning

2.3 Stefano Leonardi on $(1+\epsilon)$-Approximate Incremental Matchings in Linear Time

Definition of maximum matching

Model: graph changes over time, maintain a large matching with little overhead

Gives related work for variants

Here study problem where graph is presented edge by edge

Result

In bipartite graphs a $1+\epsilon$ deterministic approximation in amort. $O(1)$ time.

Basics

An augmenting path is a path starting and ending with two free nodes (i.e. unmatched nodes) Switching every edge in an augmenting path increases the number of matches by 1

Useful lemmas from matching theory

No augmenting path of length $2k-1$ implies a $1+\frac{1}{k}$ approximation

Outline

1. Warm-up (3/2, $O(1)$) approximation
2. Details for bipartite graphs
3. Briefly for general graphs

1. Warm-up (3/2, $O(1)$) approximation

If an arriving edge (u, v) is free, add it If u free and v matched with w; if w has a free neighbor match it and take (u, v) into your matching

Necessary data structures to implement this: A list L_v of free nieghbors; a pointer of free vertices to their entries in L_v When (u, v) is inserted, do nothin if u and v are matched; add u to L_v if u free and v matched and no aug path; if u matched update the list and pointers

Lemma: amortized running time is $O(1)$ per edge

Bad news: Searching for a 5-path can cost $O(n)$ time Good news: finding all 5-paths is sufficient but not necessary to give a $\frac{4}{3}$ path.

General idea: algorithm computes a matching M there exists a maximal set P of disjoin augmenting paths of length 5. For each $o \in P$ find k unique matching edges in M. This gives a $\frac{4}{3}(1 + \frac{1}{k})$ approximation.

3 May 24, 2018

3.1 Yuval Rabani on Lipschitz extension from finite subsets

Two metric spaces, (X, d_x) and (Y, d_y). And $T \subseteq X$. $||f||_{Lip}$ is the constant expansion of $f : X \to Y$.

$F : X \to Y$ is an extension of $f : T \to Y$ iff $F|_T = f$

Main result in paper

For every $n \in \mathbb{N}$ there is a metric space (X, d_x), $T \subseteq X$ with $|T| = n$, a Banach space, $(T, ||\cdot||_Y)$ and $f : T \to Y$ with $||f||_{Lip} = 1$ such that the extension of $F : X \to Y$ of f has $||F||_{Lip} = \Omega(\sqrt{\log n})$.

"A worst case result for extension of mapping metric spaces into Banach spaces"

Not interested in how much distances shrink; only the worst case expansion

Improves a known LB from [JL '84] (they had $\sqrt{\frac{\log n}{\log \log n}}$)

Best known upper bond is $O(\log n / \log \log n)$ None of this has any relation to "ARV"

Result uses intuition and tools from approximation algorithms (hence giving talk at FND)

This is a "recycled" result; it didn't work for the original purpose that it was intended

Connection to Approximation Algorithms

Problem of 0-extension

Input: finite graph $G = (V, E)$, $T \subseteq V$, metric d_T on T

Output: map $F : V \to T$, identity on T

Objective: minimize $\sum_{(x,y)\in E} d_T(F(x), F(y))$

"If and edge between two vertices would like to map them to close (or the same) terminal"

NP hard in general but can be relaxed

Two relaxations:

1. Metric relaxation: instead of assigneing nodes to terminals, extend the metric on the terminals to a metric on all the nodes; minimize sum of distances between vertices under this metric (really only a semimetric needed)

2. Earthmover relaxation: instead of optimizing over all metric, optimize over a subset of metric that can be efficiently optimized over

(Under the unique games conjecture this gives the "best possible result")

Define r-magnification of (X, d_x) at $T \subseteq X$ as: $d_{X,r,T}(x, y) = d_X(x, y) + r \cdot |\{x, y\}| \cap T$

"Sort of pulling out and expanding the set T; like putting a magnifying glass on the set T"

R_0^X is the set of all X-indexed real vectors v with $\sum_{x\in X} v_x = 0$

Define a norm on this set: $||v||_{w_1} = Tran(v^+, v-)$ where $v^+ = \max v, 0$ is the "positive part" of v and v^- is the negative part.

Expanders

d-regular graph $G = (V, E)$, $|V| = n$, d_G denotes the shortest path metric on V

Edge expansion $\phi(G)$ is the max ϕ s.t. for all non-trivial S it holds that $|E(S, V \setminus S)| \geq \phi \cdot |S| \cdot (n - |S|) \cdot |E|/n^2$ "Every cut approximately is random in an expander"

Some properties:

- $\phi(G)$ "embedding an expander into l_1 then the expanded distance between any two nodes is always greater than the expected length of an edge up to ϕ [NRS '05]

"If ϕ is large no way to embed an expander such that the average distance between vertices is larger than their edge (or maybe the other way around–I missed this)"

- Any two $A, B \subset V$ there are alot of edge-disjoint paths between A and B [Menger '27]
- For any subset of nodes, the expected distances between nodes is relatively large

The Construction of the Lower Bound

Fix d and ϕ, take a family of d-regular n-node ϕ-expanders

Choose a $T \subset V$ with $|T| = \Theta(n/\sqrt{\log n})$

X, d_x is the r-magnification $(V, d_{G,r,T})$ of (V, d_G) at T for $r = \Theta(\sqrt{\log n})$

"think of attaching a path of length $\Theta(\sqrt{\log n})$ to each $t \in T$"

Target space: $(\mathbb{R}_0^T)$ (What is an isometry? A mapping between metric spaces that preserves distances.)

Intuition: integrality gaps - earthmover: $O(1)$, metric $\Omega(\sqrt{\log n})$

"If you take this r-magnification metric, that's a feasible solution for the metric relaxation with cost at most $\mathrm{OPT}/\sqrt{\log n}$. On the other hand with the earth-mover relaxation this behaves well; i.e. it has a constant factor integrality gap... The distances in the r-relaxation inflate"

Gave the proof of the theorem

3.2 Chaitanya Swamy on Approximation Algorithms for Distributionally Robust Stochastic Optimization

<u>Two-stage stochastic optimization</u>

Given probability distribution over inputs

Stage 1: Make some advance decision

Observe the actual input

Stage 2: Take recourse, paying a recourse cost

Choose stage 1 to minimize stage 1 + recourse cost

<u>2-Stage Facility location</u>

Distribution over clients gives the set of clients to serve

In stage 1 can open some facilities for a certain cost

Then observe clients

Then potentially open more (expensive) facilities

Want to decide what facilities to open in stage 1 $x \in X$

Total cost, $c^T x + E_{A \subseteq \mathcal{C}}$Stage 2 cost for A

$g(x, A) :=$ optimal stage 2 LP-cost for A given x

<u>How is this probability distribution specified?</u>

1. Explicitly list scenarios (but limits us to polynomial-sized distributions)
2. Independent probabilities that each client exists
3. A black box that can be sampled (this is the setting for this talk)

<u>Incorporating Robustness</u>

Assumption in 2-stage optimization: scenario distribution p is known precisely.

Issues and criticism:

- Why is this reasonable? Seems optimistic. 2-stage optimization and overfit to p

Is there a more robust approach? In the robust setting: minimize $c^T x + \max_A g(x < A)$

Want: middle ground between stochastic and robust optimization

<u>DR 2-stage stochastic optimization</u> Collection $\mathcal{D}$ of distributions over scenarios.

Stage 1: Make some advance decisions

Observe the actual input

Stage 2: take recourse

Choose stage 1 decision to minimize Stage 1 cost + $\max q \in D\, \mathbb{E}_{A \sim q}$[Stage 2 cost for A]

Other names: ambiguous stochastic optimization

This sort of interpolates between these two extremes: either one distribution or like singleton distributions

Model dates back to Scarf (1958)

Essentially no polytime, approximation results when $\mathcal{A}$ is discrete (e.g. in facility location) Past results mostly when $\mathcal{A}$ is continuous, convex

Choices for $\mathcal{D}$:

1. Given some moments our bounds on the moments and then $\mathcal{D}$ is all distributions that meet these moment contrains (e.g. fix expectation)
    - Work in CS on correlation gap: approximation results for DR 2-stage discrete optimization with fixed expectations. $g(x, A)$ admits cost-sharing schemes
2. Ball around a central distribution $\bar{p}$ in some probability metric (what he's focusing on); feels like robust optimization stuff like Zico's
    - Popular metric is Wasserstein distance (earthmover/transportation dist.):

<u>Focus of Work</u>

DR optimization where distribution is loose as per Wasserstein metric L (two distributions are close if they "spread" probability on "similar" scenarios)

L_1 is special case where $d(A, A') = 0$ iff they're the same

In FL could define metric on scenarios based on where clients and facilities are located

Except for set cover get approximations within $O(1)$ of what is known in deterministic setting

Only assume some information regarding maximum distance between scenarios

Most interesting result: relate approximability of DR problem to approximation of π (their problem).

<u>Theorem</u>

Given

1. a η-approximation for π (pseudo-approximation note...)
2. local ρ-approximation for underlying standard 2-stage probem (rounds fractional first-stage x blowing up stage-1 cost, stage-2 cost of each...)
3. α-approximation for underlying deterministic problem

We can obtain a $O(\alpha\beta\rho + \epsilon)$-approx. for DR problem in time $poly(\text{input size}, \lambda/\epsilon)$

Often follow from known results. The new thing here is need a β-approximation for π. In various cases utilize fresult for underlying k-max-min problem.

Two main components

1. Sample average approximation (SAA): sample N times from $\bar{p}$ to construct estimate $\hat{p}$. Solve the DR problem. Show that that works here.
2. Solving SAA problem: problem with poly-size central distribution $\hat{p}$. Quite challenging: fractional relaxation fives LP with exponential number of variables and constraints. Even computing the objective value can be NP-hard.

Open Question

Results for other choices of $\mathcal{D}$

- Distributions satisfying moment conditions
- Black-box distributions

3.3 Andrew McGregor on Sketching as a Tool for Graph and Combinatorial Problems

Graphs encode structural information

Can't use classic algorithms on very large graphs

What are new 'textbook' algorithms for modern massive graphs?

Two Main Graph Stream Models

1. Insert-only model
2. Insert-delete model

Streaming

Goal: use small memory

Classic result: estimate statistics

Graph Streams: growing body of work

Sketching

Random linear projection $M : \mathbb{R}^n \to \mathbb{R}^k$ where $k \ll n$

Many results. Also parallelizable.

Today's Focus

Matchings in a graph.

Survey recent results on matchings in a) general and planar graphs and b) insert-only and insert-delete streams.

Insert-Only General Graphs: Result for Maximum Cardinality Matching

Open question of how to beat greedy algorithm. Best lower bound is about 1.582.

Weighted: $2+\epsilon$ approximation in $\tilde{O}(n/\epsilon)$ space [Paz, Soda 17]

Weighted to unweighted reduction that only loses a factor of 2.

So not going to talk about weighted matchings anymore.

Insert-Delete Streams

Exact: if matching size $\leq k$ can solve exactly in $\tilde{O}(k^2)$ space

Approx: can find t-approx matching in $\tilde{O}(n^2/t^3)$ space.

Both results are optimal. Main idea is sampling.

How to do it: a false start

Uniformly sampling edges is a bad idea... Gave a graph showing this

SNAPE Sampling

So not going to uniform sample but rather do Sample-Node-And-Pick-Edge Sampling.

Sample each node with probability $p = 1/k$ and delete rest. Pick a random edge among those that remain.

Theorem: if G has a max matching size k then $O(k^2 \log k)$ SNAPE samples will include a max matching from G.

(Missed the rest of the talk)

3.4 Anupam Gupta on Beating the factor of 2 for k-cut in FPT time

Along the lines of fine-grained approximation algorithms

k-cut

Given graph, cut graph into at least k components

NP-hard, a n^{k^2} algorithm [Hochbaum 94]

Thorup did this deterministically

Can you get k out of exponent?

No; it's W[1] hard with parameter k (you can't hope to beat n^k)

Approximations

2-approx [Vazirani 95]

$(2-\epsilon)$-approx disproves Small Set Expansion hypthesis [Manursangsi 17]

Can we do better?

Don't expect to beat $n^{O(k)}$ exactly

Can't beat 2-approx...

Can we get a $(1+\epsilon)$-approx in FPT time $f(k) \cdot poly(n)$?

We don't know but a 1.8-approx in FPT time. Today will give a 1.9999-approx

At least as hard as clique (can't find a clique in $n^{\omega k/3}$) time. Best exact algorithm take $O(n^{2k-2})$ time

Theorem

Exact algorithms in $n^{2(\omega k/3)}$

FPT Approx

Greedy algo is 2-approx

If greedy is better we're done so look at instances where greedy does poorly (bad examples have special structure) and use that structure

Proof that Greedy is 2-approx

Algo: find a min cut repeatedly $|\delta S_1^*| \leq |\delta S_2^*| \leq \ldots \leq |\delta S_k^*|$ (OPT) $2\,\mathrm{OPT} = \sum_i |\delta S_i^*|$ (Each edge is counted twice) δS_1^* is a possible cut so $|C_1| \leq |\delta S_1^*|$

This is tight; gives example

Idea 1: branching

What about $|C_i|$ vs $|\delta S_1^*|$?

Suppose $|C_i| > |\delta S_1^*|$ then δS_1^* must be completely cut (i.e. union of some algo's components is exactly S_1^*)

So guess all subsets of components. Branching factor: 2^k, branching depth: $k \implies 2^{k^2}$ time

Henceforth assume $|C_i| \leq |\delta S_1^*|$

Idea 2: if there's a gap, we win

Draw a line at $(1-\epsilon)\delta S_1^*$. If most of our cuts are below this line then you're winning. Similarly, if OPT stays above this line you win.

So bad case is when you're cuts and OPT's cuts are close to $(1-\epsilon)\delta S_1^*$.

Idea 3:

Consider all $(1+\epsilon)$-min-cuts in G

If two of them cross then $\min 4-cut \leq 2(1+\epsilon)mincut(G)$

Greedily take min-4-cuts: pay $2(1+\epsilon)$ mincut(G) get 3 new pieces

So paying about 2/3 mincut(G) per new component

So in the hard example near-min-cuts don't cross (laminar family)

Only $2^{2(1+\epsilon)}$ man near-min cuts. Can represent by a tree T.

Optimal cut = deleting $k-1$ incomparable edges in this tree

Idea 4: special algorithm for this "laminar cut" problem.

Take cuts corresponding to $k-1$ incomparable edges in this tree

Special case: partial vertex cover

Given graph H, cut out k vertices; minimize the number of edges that touch these vertices.

Exactly solving this is as hard as clique

Algo: pick k min-degree nodes Case 1: OPT $\geq k^2/\epsilon$ (overcount by only a little) Case 2: OPT $\leq k^2/\epsilon$ Randomly color edges red/blue, throw away red edges. w.p $2^{-k^2/\epsilon}$ have all intra (missed the rest here)

3.5 Thomas Rothvoss on A (1+epsilon)-Approximation for Makespan Scheduling with Precedence Constraints using LP Hierarchies

Problem

A classical scheduling problem

n jobs all with size 1 and minimize the makespan on m jobs. Some arbitrary precedence constraints.

A 2-Approximation

"The list scheduling algorithm"

Algo: pick maximal set of jobs that don't depend on any jobs. Then repeat for next time unit

Analyis: there is a **chain** active at all non-busy times. Length of chain $\leq$ OPT. Length of busy times $\leq$ OPT.

Can also shave of a $\frac{1}{m}$ factor

A LP Forumlation

Gave LP: $x_{jt} = 1$ if run j at time t

Integrality gap of $2 - \Theta(\frac{1}{m})$

Main Theorem

$2 - \Theta(1/m)$-apx in poly-time No $(2-\epsilon)$-apx under variant of UGC [Scensson '10]

Open problems: Is this problem with 3 machines NP-hard? [Gary and Johnson '79]

Main result: the integrality gap is $\leq 1_\epsilon$ after $\log n^{O(\log\log n)}$-Sherali Adams (running time of $n^{(\log n)^{O(\log\log n)}}$)

Sherali Adams

Basic idea. Some polytope relaxation K which is the integral hull of the polytope. So introduce some extra variables such that if you project out the extra variables onto the original space you get a tighter relaxation (tighter towards integral hull). Hopefully eventually converge to convex hull of integer points.

Round-t Sherali-Adams Relaxation

Given $K = \{x \in [0,1]^n \mid Ax \geq b\}$

Introduce variables for $I \in [n]$, $|I| \leq t$

$y_I \equiv \bigwedge_{i \in I}(x_i = 1)$

"Introducing joint distributions over the probability that a group of variables is 1"

Can write explicit constraints so that y looks consistent on any t variables

Inducing on one variable

Lemma: for $y \in SA_t(K)$, $t \geq 1$, $i \in [n]$, $y \in conv\{z \in SA_{t-1}(K) \mid z_i \in \{0,1\}\}$

"Can write a given point in SA_t as a CC of two points in SA_{t-1} where that variable is 1 in one and 0 in the other"

Partition

Partition time horizon into binary family consiting of $\log(T)$ families

(Missed a bit here)

The Algorithm

1. Apply conditioning until max chain length of jobs in first $O(\log\log n)^2$ levels down to $\epsilon' T$ ("long chains are a problem")
2. Among first $O(\log\log n)^2$ levels pick $O(\log\log n)$ of consecutive middle levels and discard jobs
3. For each interval I directly below middle levels: recursively call algorithm I with $\{j \mid sup(j) \subseteq I\}$ (valid schedule for bottom jobs)
4. Schedule top jobs

3.6 Andreas Emil Feldman on Parameterized Approximation Schemes for Steiner Trees with Small Number of Steiner Vertices

Steiner Tree

G with edge weights and terminals $R \subseteq V$.

Find $T \subseteq G$ connecting all terminals of R minimizing weights.

Cobham-Edmonds Thesis

Feasibly solving a problem requires an algorithm with two properties:

1. accuracy
2. efficiency

But ST is NP-Hard Solution: give up accuracy or efficiency

Giving up on accuracy=approximation algorithms

Theorem: can get a poly time $(\ln 4 + \epsilon)$ approximation

But problem is NP-Hard

Giving up on efficiency=parameterized algorithms

FPT algorithm: runtime $f(k)n^{O(1)}$ for function f independent of n

XP algorithm: runtime $f(k)n^{g(k)}$, for functions f and g independent of n

Steiner Tree and XP and FPT

An FPT algorithm for ST with parameter of tree-width [Cygan et al. 11]

An FPT algorithm for ST with parameter of $|R|$ [Dreyfus '71]

An FPT algorithm for ST with parameter of vertices $|V \setminus R|$ [folklore]

An XP algorithm for ST with number of vertices in the optimum solution [folklore]

Gave some experiments on supercomputer and laptop to justify why we should care about XP and FPT algorithms

Hardness of ST

Problem is APX hard; i.e. no PTAS unless P=NP

ST is W[1]-hard for parameter of number of vertices in the optimum solution

So need to give up on both so they give an APX FPT algorithm

Technique comes from

Preprocessing Algorithms

1. Remove "easy parts" to obtain "core"
2. solve "core"

e.g. for ST remove a steiner vertex if it is very far from any terminal then just solve problem on remaining graph

Definition: an α-approximate preprocessing algorithm $\mathcal{A}$ for parameter k is a pair of polynomial time algorithms:

- reduction alg: given instance (I, k), compute new instance I', k'
- lifting alg: given β-apx to (I', k'), compute a $\alpha \cdot \beta$-apx to (I, k)

If $|I'| + k' \leq f(k)$ then $\mathcal{A}$ is an α-approximate kernel

Lemma: a problem with parameter k has a parameterized α-approximation iff it 'has a kernel'

Reduction algorithm

Add a tree to solution with

1. simple structure (star)
2. small weights
3. many terminals

Definition: $w(S)/(|V(S) \cap R| - 1)$

The -1 biases towards larger stars

Lemma: a star of smallest ratio can be found in polynomial time

Their Algorithm Find smallest ratio star, contract it until number of terminals is small. Solve remaining instance using FPT algorithm. Uncontract all stars in the solution computed by FPT algo.

Analysis

Stars charged to optimum: add star to optimum; remove edges in cycle in optimum (this is what we charge star to)

Gave problem example for charging stars to deleted vertices up to $(1 + \epsilon)$

Rest of the talk is about how to charge the bad stars to the optimum (main technical contribution)

4 May 25, 2018

4.1 Sylvia Boyd on The Salesman's Improved Tours for Fundamental Classes

Metric TSP

Complete graph on n nodes with a metric (satisfying triangle inequality)

Let OPT(c) be cost of min Hamiltonian cycle

Natural LP Relaxation

Subtour LP

$x \in \mathbb{R}^E$

Constraints basically say for each subset of edges if you look at xs in the cut it has to be at least 2 (= 2 if it's a single node)

Solutions to this LP is often sparse so look at support graph, G_x

$LP(c)$ lower bounds OPT(c)

Integrality Gap of LP

In practice this lower bound is good but what about theory?

Consider integrality gap α of this LP

$4/3 \leq \alpha \leq 3/2$; no improvment in over 30 years!

4/3 conjecture is that integrality gap is exactly 4/3

In this work they show $\alpha \leq 10/7$ (between 4/3 and 3/2) for an "interesting" class of cost functions

More important though, they get new tools

So take **polyhedral approach**; take subtour LP polytope, S^n

Instead of ham cycles they consider tours: edge-set of a spanning Eulerian multi-subgraph of K_n

Incidence vector of $\chi^J \in R^{E_n}$

Equivalent statement of 4/3 conjecture [Vempala '04]: for any $x \in S^n$, $4/3x$ is in the convex hull of the incidence vectors of tours

Fundamental Classes

In fact, there are classes of points in the polytope that are sufficient to prove that overall this is true

These points are called fundamental classes: showing that ρx is in the convex hull of tours for all $x \in F$ implies the same holds

E.g. B-C points: set of all $x \in S^n$ satisfying [Carr '11]

1. Support graph G_x is cubic
2. Fractional edges of G_x form disjiont 4-cycles
3. In G_x there is exactly one 1-edge (i.e. a fully integral point) incidence to each node

Try to improve the 3/2 bounds for the simplese form of the B-C where all fractional values are 1/2 (so 3 kind of irrelevant here)

Main Result

10/7 for a superclass of 1/2-integer B-C points Square points: 1/2-integer B-C points with 1-edges replaced with path...

Warm up

GA 1-tree is a set $F \subseteq E$ s.t.

1. $|F \cap \delta(1)| = 2$
2. $F \setminus \delta(1)$ forms a spanning tree on $V \setminus \{1\}$

Take 3/2 proof and build on it with two ideas

1-tree polytope of G: the convex hull of 1-trees of G

But want a convex combination of tours: use a T-join where $T =$ *odd* degree nodes in the 1-tree

Odd-join polyhedron for a graph G: the dominant of the CC of odd-joinds of G which is given by $\{y \in R^E, y(C) \geq 1 foreachcutC \ldots\}$

In essence get x as a CC of 1 trees and $x/2$ as a CC ...

Add two Ideas

Idea 1: Suppose trying for $4/3x$. Try to find a "good" tour T for our convex combination for $4/3x$, take a fraction of it way and hope we can get the rest as a CC of tours. $4/3x = \lambda\chi^T + (1-\lambda)x^*$

A good tour is a Hamilton cycle.

Theorem: for an y square point x, the support graph G_x has a Ham cycle that contains all 1-edges

Magically choose $\lambda = 1/3$; i.e. take 1/3 of Ham cycle and 2/3 of "the rest"

Is x' feasible for S^n? Not always. Call these *bad* cuts.

What if there are bad cuts? Need another way of getting x^* as convex combination of tours.

Idea 2: for the CC of 1-trees for the square point x, have the 1-trees satisfy some additional useful property which will make them *good*. In particular want them to use an even number of edges across every bad cut. If have an even cut "don't need to use anything for the join". Use idea of *rainbow 1-trees*. A 1 tree in which every edge has a different color according to some edge coloring; forms the basis of partition matroid and 1-trees (both bases of matroids)

Nice Theorem: Let x be a 1/2-integer point in S^n and let $\mathcal{P}$ be any partition of the 1/2-edges into pairs. Then x is in the convex hull of 1-trees that each contain exactly one edge from each pair in $\mathcal{P}$

Defined "sad" cuts (missed this)

Theorem: 4/3x is in the convex hull of tours

Proof: express the rest as x plus T-join vector for each 1-tree

What if both bad and sad cuts?

Somehow by "balancing" two previous ideas can get 10/7

A Few More Results

1. Hamilton cycles form a delta-matroid. Can use it to get a poly-time aglorithm
2. Other classes of interest
3. Another use for Ham Cycle of G_x

4.2 R Ravi on Applications and Constructions of Cut-Covering Decompositions for Connectivity Problems

TSP problem

2EC: find min weight 2-edge connected subgraph of G; different because can be nodes with odd degree

Subtour elimination LP given

Note: if edge weights obey triangle inequality, can delete degree constraints and get same bound

Tours and shortcuts

A connected Eulerian multigraph gives a TSP solution (b/c can just shortcut over cycles)

Gave 4/3-conjecture Can also replace TSP with 2-edge-connected spanning multigraph

An α-vector of $G = (V,)$ is a vector where $v_e = \alpha$ for all $e\ inE$

The $2/(n-1)$-vector of K_n is in the subtour polytope. Proof is just algebra.

Will try and look at α-vectors in graphs and ask if it can be written as a CC of tours. E.g. in K_4 can decompose 2/3 uniform cover into 4 tours

Care about this b/c can restate 4.3 conjecture in terms of this; if for any k a $8/3k$-uniform cover on any k-edge-connected k-regular graph then get 4/3 conjecture.

What is known about these uniform covers?

- For any k can get a $3/k$-uniform cover
- When $k = 4$ can get a 2/3-uniform cover for 2EC (not constructive)
- When $k = 3$ can get a 7/9-uniform cover for 2EC (not poly time)
- Sylvia's talk showed $k = 3$ then get a 6/7-uniform cover
- This talk: $k = 3$ gets 18/19-uniform cover (worse than previous but now poly time)

Theorem: Let G be a bridgeless and cubic, then G has a cycle cover C that covers all 3-edge and 4-edge cuts of G

"Small cuts are covered across by this cycle cover"

Combine two solutions each of which is a CC of tours to get promised load on every edge. So just need to show that each of the two things combined is a CC of tours.

Use 3 different types of edges in these two sets C, M and F. C is a cycle cover given by above theorem. M is edges when you contract C. F is the stuff you tossed out with M.

First one

Put 1 in C and 4/5 on M. Contracted graph is 5-edge connected so putting 2/5 on every edge puts you in the polytope.

Second one

"Only put halves on cycles and cycles cross cut an even number of times"

Other Result Below 1

There is an 15/17-uniform cover for 2EC on 3-edge-connected cubic graphs

Theorem: a 1/2 integer solution to tape then 4/3 of that can be decomposed into integral feasible solutions to TAP. Did a similar CC decomposition here

Second CC is most delicate part of talk. Put 1/2 on C and 1 everywhere else. This is in connector polytope. End up with a TAP solution and then apply the above theorem to multiply by 4/3.

More Results

Get even better results for bipartite graphs Node weighted version of TSP; get better results

Open Problems

- Better than 3/4-uniform for 4-edge-connected of 4-regular graps
- Improve 18/19 on 3-edge connected cubic graphs
- One other one

4.3 Mohammadhossein Bateni on Efficient Routing for Online Shopping Services

Online delivery motivation: being the middle-man delivery man has opened a new market

Problem

Weighted G, (s_i, d_i) deliver demands. Find minimum length tour of a vehicle of capacity k that satisfies all demands.

Variants:

- Planar graph
- Demand pairs share source
- Tour or path (doesn't matter)
- Rooted or unrooted (doesn't matter)
- Large vehicles with no capacities (necessary; the problem they work on)

With UGC hardness is $(2 - \epsilon)$

Dial a ride without the hardness of UGC is $1.5 - \epsilon$

Hardness

Hardness reduction is from **minimum feedback vertex set**: given a directed graph remove as few vertices as possible to get an acyclic graph.

Form a star with a root r connected to all original vertex with edge weights set to 1/2. Add a demand for every edge in the original graph. Gave directions of reduction.

Simple Algorithm

Find TSP tour on sources. Then find another tour on sinks. If TSP has ρ-apx then get a 2ρ-apx. This analysis is tight.

Revised Algorithm

Find TSP at r of all sources and destinations in first round. In second round only go to remaining destinations. Also a 2ρ-apx.

Gives results of ρ for varios kinds of graphs (e.g. planar)

Can we do better than 3-apx?

Assume $O(1)$ sources and many destinations. The simple algorithm in this case gives $1 + \rho$ approximation.

How else can we improve? Can solve optimally on tree.

Road networks look very similar to planar graphs

PTAS Framework

1. Start with constant factor APX
2. Add some more edges to construct a spanner (uses TSP spanner construction [Klein '08, BorLeWul '17])
3. Treewidth reduction via "shifting" ([DemHajMoh '10, DemHajKaw '11])
4. Solve bounded-treewidth instance (tircky; guess order of visiting sources; no unique path between sources; use tree-width decomposition; run dynamic programming on decomposition)