IPCO 2019

David Hershkowitz

May 29, 2019

These are the notes I took while at IPCO 2019. I was not able to make the morning sessions on the second day or afternoon sessions on the last day. These notes were written while trying to keep up with the talks and so are not free from errors. Cheers!

Contents

1 May 22, 2019

1.1 Strong Mixed-integer Programming Formulations for Trained Neural Networks (presented by Joey Huchette)

Paper link: https://arxiv.org/pdf/1811.08359.pdf

Use IP to solve a problem with a NN embedded in it

The Problem

In standard supervised learning task want to learn some function from historical data $(x_i, y_i)_i$. Want good generalization: on unseen point (x^*, y^*) would like $NN(x^*) \approx y^*$

In optimization would like to maximize some function f. Here could try and instead maximize a neural network output x. Hard because nonconvex and combinatorial constraints

Application: Veryifying Robustness

Problem with NN is small perturbations in input can cause NN prediction to be wrong; i.e.\ NNs not locally robust

Proving or disproving robustness of a NN is an active field of ML research

Application: Deep RL

Prediction problem: Want NN output to approximate the RL $Q(x, a)$ function

Optimization problem: Optimize over policies

Application: Designing DNA for Protein Building

Prediction problem: probability of DNA sequence binding to a protein

Optimization problem: find the best binding sequences

The Solution

Can look at primal side or dual side to optimize; in middle have mixed integer programming

Contributions

MIP formulations for popular NNs; stronger than existing approaches

- Most common case: affine function on box domain
- Structure: tight big-M formulation + efficiently separable cuts

Also some computation results

MUP Formulations for NNs

A MIP formulation is a polyhedra…

Looking for an ideal formulation

NNs = Piecewise Linear Functions

Complexity of NNs comes from composing together many simple non-linear functions (e.g. ReLUs)

Ideal Formulation for ReLU Neuerons

Gave an ideal formulation for NNs

Look at each subset of the variables and write some "big-M" formulation. Exponential number of constraints but separable

How Do We Get Here

1. Write down ideal “multiple choice”’ formulation
2. Rewrite all logic as bounds on output y (a primal optimization); this is ideal but still not linear; changing it again makes it “hereditarily sharp”
3. Apply Lagrangian relaxation to aggregation constraints
4. Analyze further

Computational Results

Train a small network; with L1 regulation speeds up their result

Q-Learning

Are optimal actions tractable?

Some preliminary results where their approach outperforms gradient descent

Conclusion

MIP formulations for optimizing over trainded NNs

Many applications

Framework of independent interest: recipes for strong formulations for the max of d affine functions

1.2 The Asymmetric Traveling Salesman Path LP has Constant Integrality Ratio (presented by Vera Traub)

Paper link: https://arxiv.org/pdf/1808.06542.pdf

About the path version of ATSP

ATSP Definition

Find cheapest closed walk that starts and ends at same point and visits all the vertices

Path vesion generalizes this: given a start vertex d and end vertex t where walk starts at s and ends at t

Related Work

Svensson et al. showed constant factor approximation for ATSP

Feige and Signh showed how to reduce ATSP to path version with constant overhead; though this result doesn’t say anything about integrality ratios

Main Result

If ρ is the integrality ratio of ATSP then the integrality ratio of ATSP path is at most $4\rho - 3$

Previous work gave $O(\sqrt{n})$ then $O(log n)$ish; this is a constant

Lower Bounds

Best known lower bound is 2 for ATSP (previous work) and 2 even for unweighted ATSP (as shown in this paper)

ATSP LP

Classic subtour eliminatio LP. Have circulation constraints that say every vertex has equal in and out degree as ell as subtour elimination constraint that says that $x(\delta(U)) \geq 2$ for every subset of vertices U.

Path version is similar except now require an $s - t$ flow of value 1

Feedback Paths

Start with some LP solution for path version; want to obtain an integral solution for path version of cost $4\rho - 3$ times the LP.

Want to exploit the fact that ρ is the integrality gap of the ATSP problem.

Thus, add a vertex so now can turn your path solution to a classic ATSP problem; then apply your classic ATSP rounding to get an integral ATSP solution of cost $4\rho \cdot LP$; then delete your edges; problem now is you don't necessarilly have an $s-t$ tour; so have a bunch of walks $P_1, \ldots, P_k$ from s to t that would like to merge into a single walk without increasing the cost by too much.

Merging Walks WLOG edge set of graph have nonzero LP support.

Now if we look at the strongly connected components of our graph all walks must visit all SCCs in the same order (otherwise would violate the definition of strongly connected); add shortest paths within strongly connected component to collate the walks

Why does the cost not increase by too much? Will make use of dual LP to show cost increase is no more than $3 \cdot LP$

1. Give some upper bound of $LP + 2(a_s - a_t)$; where these are dual variables
2. Argue that $a_s - a_t \leq LP$

Key difference here to ATSP is that the node potentials matter

Techniques for 1

1. Standard uncrossing to argue support of dual variables is laminar
2. Complementary slackness to show that costs are laminarly weighted (with some correction from dual variable node potentials)
3. Now want to analyze cost of $s - t$ walks. by previous point node potentials telescope so cost of $s - t$ walk is $a_t - a_s +$ sum of costs of leaving laminar cuts. So now just need to bound c^y costs of extra paths within SCCs. Can argue that dotted paths can visit each laminar family at most which upper bounds your cost of paths in terms of total dual solution (which is about LP cost)

Techniques for 2

Lemma: don't take any optimal dual solution but one where $a_s - a_t$ is minimum. In particular, for any set U can get from $s - t$ without using that set under the support of the dual solution.

Use of lemma: essentually apply Menger's theorem to show can get two paths that don't enter some set and visit every set at most once between the two of them. This allows them to upper bound the difference of a_s and a_t; I think also using the fact that there is a feasible dual solution of cost 0 here to set up this inequality

1.3 A Generic Exact Solver for Vehicle Routing and Related Problems (presented by Artur Pessoa)

Defined Capacitated Vehicle Routing Problem (CVRP)

Also many variants

Focus on exact algorithms here

Related Work Most succesful results based on column generation

Recent results used Branch-Cut-and-Price algorithm

- Concept of limited-memory cut was important
- Combined and enhanced results of previous results

Work on this got a best paper in MPC

Good News

Now possible to solve instances with up to 200 customers

Typical instances with 100 customers solvable in less than a minute

Bad News

Designing a state-of-the-art BCP algorithm needs several work-months of a specialized team

Proposal of This Work

A BCP solver for a generic model that encompasses a wide class of CRPs

Introduce idea of packing sets

The Basic Model

Parition a set of resources R into main resources R^M and secondary resources R^N

Secondary resources can be of a special type that allow "negative consumption"

Associate paths with variables

End up with LP with exponential but separable constraints

Then gave the dual

Basic model leads to a robust Branch-Cut-and-Price (BCP) algorithm

Including Advanced Elements: Packing Sets

A **packing set** is a subset of arcs (possibly from different graphs) where in each subset there is at least one optimal solution that never visits two arcs in the same packing set

Modeling Example

Gave example for Generalized Assigment Problem (GAP) where tasks are assigned to machines to minimize total assignment cost

Code

Code is available for academic use

1.4 On Friedmann's Subexponential Lower Bound for Zadeh's Pivot Rule (presented by Alexander V. Hopp)

Simplex Algorithm and Zadeh's Pivot Rule

Simplex algorithm picks a vertex and pivots to a new vertex

Zadeh's pivot rule: choose improving direction least often

Question of whether this is a polynomial time pivot rule; in '11 Friedmann gave an LP where need $\Omega(2^n)$ iterations using this pivot rule

Agenda

1. Markov Decision Processes (MDPs)
2. Lower bound construction of Friedmann
3. Their Contribution

MDPs

A graph with player vertices and randomization vertices; player chooses actions, randomization chooses

Value of a vertex is expected discounted reward

Can find an omptimal policy by performing improving switches: change action to improve value of a player vertex

MDPs can be formulated as an LP

Theorem: if you consider the dual of the LP then natural terms of dual translate into natural terms of MDP; including the pivot rule

This enables us to analyze MDPs instead of LPs when considering pivot rules

Friedmann's Idea

Goal: exponentinal number of operations

Idea: binary counting with n bits

Intuitively: build MDP s.t. Zadeh simulates n-bit counter

In this way policies represent binary numbers

How does the construction ensure that Zadeh's rule is obeyed though?

Ansatz used gadgets to represent bits: will have an edge that is **active** iff the edge is in the policy

Problem is present because less signifiant bits switch much more often than more significant bits

Friedmann said "maybe it's sufficient to not have all records balanced"; used larger gadgets with two copies of smaller gadgets; then hae only 1 copy really represent each bit; that is, copies alternate representing bit i; this allows you to have ocurrence records balanced

Gave details of bit gadgets

Friedmann's Proof of Correctness

Theorem: given policy π_b representing b can apply Zadeh's rule to get π_{b+1} representing $b + 1$

Divides application into six phases; for each phase

- determine improving switches
- determine ocurrence records
- give application order of switchs
- one other phase

Unfortunately MDP has size $O(n^2)$

Their Contribution

Described phase 3 in more detail; showed that this application of improving switches in this phase does not actually obey Zadeh's rule; moreover this general idea doesn't work

Also managed to show that can repair this proof; in particular in phase 3 swiches can be applied without violating Zadeh's rule; but this application is not "level by level" because they showed that this cannot possibly work

Summary

There is an LP of size $O(n^2)$ s.t. simplex with Zadeh's rule needs $\Omega(2^n)$ iterations

Connected LPs and MDPs to simulate binary counters; used policies to represent numbers and enumerate binary numbers

Showed flaw in Friedmann's proof and then repaired it

1.5 Linear Programming using Limited-Precision Oracles (presented by Dan Steffy)

Will talk about some of the best methods in practice for solving LPs

Outline

1. Background
2. Oracle Algorithms and Compleixty
3. Computations
4. Conclusions

Algorithms for LPs

Interior point

- Poly time
- Efficient in practice
- Well parallelized

Simplex Method

- Not poly time
- Efficient in practice

Exact LP Solutions over Rationals

Typically high precision arithmetic is used to compute ϵ-optimal solution which is then converted to exact rational solution

In practice we typically use floating-point arithmetic

From Approximate to Exact Solutions

Exact solutions can be reconstructed from high accuracy approximations

Given input α and bound B on its denominator. Want p/q such that $\alpha - p/q < 1/2B^2$

One way to do this is using continued fraction representation

Gave an example of how to use continued fraction representations; the last convergent of continued fractions with denominator less than B is the desired p/q

LP Example

Example LP where need a very small ϵ tolerance but solution is reconstructible with much less precision

Exact LP in Practice

In practice gradually ramp up precision

Iterative Refinement

Approximately solve; compute residual; then find another solution to account for residual; repeat over and over

Can do basically the same thing for LPs: solve LP approximately; get approximate solution; multiply by a scaling factor to separate out inequalities; repeat

So idea is to solve a series of LP subproblems with fast low-precision arithmetic; compute errors and corrections exactly; keep iterating until desired accuracy reached

Nice because, e.g.\, can warm start solutions

Contributions

Trying to understand the running time of this algorithm better; chose oracle model of complexity

A **limited-precision LP oracle** satisies feasibility and approximate optimality within some specified conditions

Theorem: Assuming they have some such oracle, getting a τ-optimal/feasible solution possible oracle-polynomial time

Proof Idea: each refinement round reduces feasability and optimality violations by a fixed number of digits; so only need linear oracle calls in encording length of tolerance

Theorem: given iterative refinement for LP and then every once and while solve basis system exactly this can get an optimal **rational** solution in oracle-polynomial time

Proof Idea: Exact basic solution either optimal or primal or dual violation bounded below by a function of the input data

Theorem: instead of doing basis verifications as in least theorem, do rational reconstruction as above; this can also give an oracle-polynomial time algorithm

Remarks

Both algorithms have oracle polynomial but have good potential to get lucky and terminate early

Computational Results

Compared using SoPlex with the two above roundings; doing exact rational system solves ended up faster; also compared to QSOpt

Conclusions

Introduced limited-precision LP oracles and studied complexity under this model

1.6 Min-Max Correlation Clustering via MultiCut (presented by Saba Ahmadi)

Correlation Clustering

Given positive edges of "similar" nodes and negative edges of “dissimilar” nodes

Goal: cluster nodes to minimize total number of disagreements

A mistake is if two similar nodes are in different clusters or two dissimilar nodes are in the same cluster

Problem introduced by Bansal, Blum and Chawla

If there exists a perfect clustering, it’s easy to find. Just discard dissimilar edges and let clusters be all remaining connected components

Hardness is if perfect clustering is not possible

Related Work

Two objective functions have been considered: minimizing disagreements or maximizing agreements; considered both on general and complete graphs

Minimizing disagreements is harder

These were global objectives

Local Guarantees

Now consider a local objective; namely minimize number of mistakes at the worst vertex (or do same for maximization); want to be fair to all the vertices

Gave an example of local vs global guarantees

This local version is NP-hard for general graphs; a 5-approx for complete graph

For general weighted graphs a $O(\sqrt{n})$ approximation

This Work

Defined a local **cluster-wise** objective

Goal here is to minimize the number of mistakes **at the worst cluster**

Main result: $O(\log n)$-approx for general weighted graphs; another approximation for another setting I didn't catch

Motivating Example

Suppose a bunch of cliques with similar internal nodes and disimilar external nodes. Also a vertex v with all similar edges to cliques C^+ and negative edges to cliques C^-

Natural clustering for worst vertex objective v combines v with C^+; will be very unfair to v; their objective keeps v in its own cluster

Their objective

Based on Bansal clustering ideas but differences between these two problems; e.g.\ number of clusters not fixed here and also have to pay for negative edges here

Use idea of Demaine et al. that classical correlation clustering and mult-cut problems are equivalent (multi-cut problem is to separate all (s_i, t_i) pairs with min number of edges)

To adopt these idea, they need to define a min-max version of multi-cut; reduced their correlation problem to the multi-cut problem (in a way that preserves the approximation factor) and then get an $O(\log n)$ approximation for the min-max multicut.

Min-max Multi-cut Approximation

1. (Pre processing) Guess number of partitions in optimal solution as k; set weight of each vertex to 1
2. SDP rounding to find a set of clusters such that sum of weights in clusters within a constant factor of average weight of a cluster (the set has "adequate coverage"); solve SDP relaxation; sample an n^3-orthogonal separator
3. Find a covering of all vertices; invoke step 2 to choose a set with large weight; then decrease the weights of all the vertices in this set; keep repeating until the sum of all weights is at most 1
4. Need to get rid of overlaps between sets; sort sets returned in covering set and do some stuff; show via a potential function only need a polynomial number of iterations and lose a factor of at most 2

Future Directions

- Prove hardness for min-max correlation

- Constant approximation for un-weighted complete graphs

1.7 Improving the Integrality Gap for Multiway Cut (presented by Vivek Madan)

Paper link: https://arxiv.org/pdf/1807.09735.pdf

Multiway Cut

Weighted graph and terminals $s_1, \ldots, s_k$

Want to remove minimum weight subset of edges to separate all terminals

With 2 terminals this is just min $s - t$ cut

But with $\geq$ 3 terminals the problem is NP-hard

A 2 approximation rounding the natural hitting set LP

This is tight so want a better LP

CKR introduced a better LP where every node is in a part (or multiple fractionally) and you pay the weight of crossing edges

Nice geometric interpretation: embed vertice into hyperplane where sum of coordinates is 1 and the cost of an edge is the distance between edges; were able to upper bound the integrality gap at 1.5

A series of followup papers gave a 1.2965 integrality gap for this relaxation

Lower bound of 1.0909 and then improved to 1.2

Thus question is what is the integrality gap?

Moreover, if integrality gap is α then there is no $\alpha - \epsilon$-approximation for this problem assuming the UGC

This Work

They improve the lower bound from 1.2 to 1.20016

A non-trivial 3-dimensional instance

Embedding and Non-Opposite Cuts

Show a lower bound on IG in unusual way; weaker than how one normally does it

Can assume all variables is a multiple of $1/m$ by discretizing which gives a discrete versioon of this embedding problem

The fractional value, in this version, is just $1/m \sum_i w_i$

Enforce a stricter condition where terminals have to be separated from opposite faces in geometric interpretation; harder to achieve this than multiwaycut

By a previous work this is sufficient to get integrality gap

But in 2 dimensional instances this can't get better than 1.2; so have to go to 3 dimensions

3 Dimensional Gap Instance

Intuition is to combine two instances each of which have low integrality to get a large gap. Intuition is that light cuts in one are heavy in the other and vice versa

In fact, doesn't work out with 2 instances; have to use 4 instances; first 3 instanes are $2D$ and last is uniform

Instances

1. I_1 is AMM '17 instance
2. I_2 such that each light cut in I_1 is now heavy; uniform weights on L_{12}, L_{13}, L_{23}
3. I_3 uniform weights on "green edges"
4. I_4 is uniform weight γ on all edges

Then just take a convex combination of instances

1.8 A New Contraction Technique with Applications to Congruency-Constrained Cuts (presented by Martin Nägele)

Problem Setting

Given an undirected graph with edge weights and vertex multiplicities at each vertex

Want a cut so that the sum of value of vertices inside is 0 mod r for some input r and want to have minimum such feasible cut

Motivation and Prior Results

Generalizes well-known cut problems

Connection to integer programming with bounded subdeterminants; where matrix A is m-modular

Congruency-constrained submodular minimization: efficient algorithm for prime power moduli

Result

Theorem 1: A polynomial time, randomized approximation scheme for CCMC with constant modulus m

Main idea: sample vertices a la Karger edge contraction and use splitting-off techniques from graph theory

Theorem 2: an exact algorithm when $m = pq$ for prime p and q

Theorem 3: a structural result

Karger's Contraction Algorithm

Algorithm: contract a u.a.r. edge until only 2 vertices

Main point in analysis is that $\delta(v) \geq$ OPT for min-cut.

Probability of a bad contraction is just $\Omega(\frac{1}{|V|^2})$

Natural question: why does Karger's algorithm fail if we add congruence constraint; well now singletons aren't necessarilly feasible so no lower bound on OPT; also average degree can now be small

The Plan

Can ignore vertices that are 0 mod m

Ignorethese vertices; then to Karger but not necessarilly according to edges in the original graph: namely, construct an auxiliary graph on vertices

Auxiliary Graph

Use splitting-off techniques: a way of reducing a large graph to a smaller one

Take a vertex and two incident edges, remove edges and connect endpoints; delete isolated vertex

Usually used to maintain certain connectivity properties

E.g. [Lovasz] the one they use: can split edges on an Eulerian graph such that

- cut values do no increase
- certain min cut values are preserved for vertices $q \in Q$

Get weighted version of this using algorithms of Frank

Example of Odd Cuts

After doing splitting off procedure end up with fact that singletons in the remaining graph correspond to feasible solutions

Thus get back the inequalities that Karger needs so can now do Karger in remaining set

CCMC with Prime Modulus p

Problem is after splitting off now singlenots no longer feasible

But if can find two vertices that are r mod p can construct a solution from these two vertices

Use theory of Cauchy-Davenport that among o nonzero elements there is a subset summing to r mod p

So can do a Karger-type average-degree analysis

General Modulus

Cauchy-Davenport theorem fails for general modulus

Problem is if average degree is low; but then many vertices with small degree and so many vertices with same multiplicity; leverage this to change γ value of a set without changing its cut value by much

Complete Algorithm

Gave overall algorithm using "enumeration", "contraction" and "reduction"

1.9 Lower Bounds and a New exact Approach for the Bilevel Knapsack with Interdiction Constraints (presented by Federico Della Croce)

Outline

1. Notation and Problem
2. Literature
3. Lower Bounding
4. Exact Solution
5. Computational Results

Problem

A generailzation of knapscak problem; can be formulated as a Stackelberg game

Leader first selects items not to exceed their capacity; follower packs a set of the remaining items to maximize profits; leader wants to minimize profit of follower

In BKP two different knapsack capacities C_l for leader and C_f for follower

Literature

Introduced by DeNegre in 2011

Solved in practice up to 50 items by CCLW then 55 by IJOC (proposed the "FLMS" algorithm)

This problem is Σ_2^P-complete (in PH hierarchy)

<u>Lower Bounds</u>

Assume items sorted according to bang for buck and items selected until the sum of weights is less than C; the first item that cannot be fully packed is the only fractional variable (the "criticial" variable)

In this problem only a criticall item if the sum of the remaining weights is greater than C_f

So now consider the optimal solution vector x^*; in the follower problem either

1. There is no critical item (easily handled case)
2. There is a critical item; they guess what the item is

For a given guess on this item write an lp to minimize $\sum_{i=1}^{c-1} p_i(1 - x_i)$ *Proposition*: get a lower bound based on critical item

Then improve the bound: extend first model to a second model with a factor assuming can add "tuples that can improve the solution"

<u>New Exact Approach</u>

1. Handle the non-existence of a critical item
2. Identify relevant critical items
3. Construct model and solve linear relaxation
4. Use standard techniques to eliminate solutions
5. Explore each relevant subproblem

<u>Computational Results</u>

"Strongly" outperforms previous algorithm on some benchmarks

1.10 A Bundle Approach for SDPs with Exact Subgraph Constraints (presented by Elisabeth Gaar)

<u>Overall Picture</u>

- Take NP-hard problem
- Take SDP bound
- Use exact subgraph constraints to improve the bound
- Reformulate the SDP
- Use the bundle method to solve the problem
- Get strong bounds, fast

<u>NP Problems Considered</u>

1. Max cut
2. Coloring problem: want to know chromatic number of a graph
3. Stable set problem: focus of this talk

Compelexity

.878 approximation for max cut

Stable set and coloring are hard to approximate

Want: efficiently computable lower/upper bounds; can use the bounds in a Branch-and-Bound algorithm

Stable Set Upper Bound

Well known that the independence number of a graph is at most $\theta(G)$, Lovasz's theta function

θ computable by an SDP

Want to strengthen $\theta(G)$ to get better bounds on $\alpha(G;)$ will use exact subgraph constraints

The Squared Stable Set Polytope

Will look at the convex hull of all stable set vectors times the transpose of all stable set vectors

If we use original SDP formulation but then include that X must be in this convex hull then get back α

But this new constraint is not polynomial

Exact Subgraph Constraints

Restricting problem to subgraph yields smaller instance of the same problem <-> downward monotone

Idea is that whenever we have X in the above convex hull then same is true if we look at a subgraph

Idea is to add that this is true for small subgraphs; can build a hierarchy based on the size of the subgraph

Exact Subgraph Hierarchy

Some computations showing that their hierarchy converges

Improving on this Idea

Don't include all subgraphs of a particular size; question then is how to choose subgraphs?

Basically a separation problem: search for violated subgraphs

Also need to solve this new SDP: need many new constraints and variables

Will adapt the bundle method to solve these SDPs

Bundle Method

Have a current center $\bar{y}$ the bundle, a set of triples

In each iteration minimize not the function h but a subgradient of function plus a penalty charge that forces us to stay close to the current center; then update

Applying the Bundle Method

Reformlate; "shake it and the right thing comes out"

Computational Results

The more exact subgraph constraints the slower Mosek gets (it's super slow); bundle is good to get close to optimal but getting really close takes a lot more time; not such a problem for their purposes

Additional computational results for stable set

1.11 Random Projections for Quadratic Programs over a Euclidean ball (presented by Leo Liberti)

Random Projections

Have a data matrix A; would like to do clustering on columns but they have high dimensionality; so premultiply matrix by a random projection into a smaller space; distances are kept almost invariant

This invariant happens "with arbitrarily high probability"

JL Lemma

There is a $k \times m$ matrix T that maintains Euclidean distances up to an ϵ

In practice would have to resample T a few times; in practice he only samples it once because would only incur a few mistakes

k is also independent of the original number of dimensions m

2 May 23, 2019

2.1 Extended formulations from communication protocols in output-efficient time (presented by Manuele Aprile)

Several problems in comb opt can be formulated as LPs

If polytope has only a few constraints can solve efficiently in theory + practice

But many times have exponentially-many constraints

Can try and add some variables and reduce the number of constraints: an **extended formulation**

Gives a geometric example where projecting into higher dimension reduces the number of facets

Slack Matrix

A way of constructing an extended formulation: a row for each facet and a column for each vertex; want a factorizaiton of this matrix

Yannakakis: if can factor slack matrix S into TU with T, U nonnegative then can get an extended formulation with small number of inequalities; but many equations

So really want a formulation where number of equations is also small

Finding such a formulation is an issue for e.g.\ knapsack, STAB

Deterministic Communication Protocols

A way to attain factorizations and EFs

Have players Alice and Bob where Alice gets row x and Bob gets column y; want to compute $M_{x,y}$ by exchanging as little information as possible

Can jointly search until know that they are in some submatrix where all entries have the same value (i.e.\ monochromatic rectangles)

Complexity of the protocol is the number of bits / height of the communication tree; want to know the number of rectangles

This partition of the matrix gives a nonnegative factorization of the slack matrix

Thus, deterministic protocoal implies factorization of the slack matrix implies extended formulation

Questions is can we skip the factorization step and get directly to the extended formulation

In this paper, show yes

Results

- General efficient techniques to turn deterministic protocols into extended formuations
- Apply to STAB
- Extend to general protocols

Algorithm Sketch

Replace node in communication tree with an intersection / convex hull of polytope corresponding to each rectangle

Thus, need description of polytopes corresponding to rectangles: these systems are simple because we don't have a T matrix (constraints on nonnegative variables)

Theorem: Can construct an extended formulation for size linear in the sum of the formulations of the leaves of the rectangles

STAB(G), G perfect

An application of their algorithm

G is perfect iff it is in the STAB(G) polytope; exponential number of inequalities

There is an SDP of small size but the smallest LP is of quasipolynomial size

Algorithm is as follows. Take set of special vertices; take graph induced by neighbors of special vertices; have to take complement of some of these graphs; do this until all subgraphs are of, say, constant size; then take exact formulation of subgraphs; then combine these together as before

Efficiency of algorithm depends on depth of this decomposition: show that by choosing low degree vertices if many low degree vertices or otherwise taking the complement of the graph suffices to get low depth

Theorem: This gives an EF of quasipolynomial size in quasipolynomial time because halve the size of the graph at each level

Unambiguous NonDeterminsitic Protocl

Can adapt these protocols but with quasipolynomial time as opposed to linear overhead as aboev

Open Questions

1. Can this be extended to random protocols
2. Is there a polynomial size EF for STAB(G), G perfect

2.2 On perturbation spaces of minimal valid functions: Inverse semigroup theory and equivariant decomposition theorem (presented by Yuan Zhou)

General Purpose Cutting Planes from Cut-generating Functions

Consider a row from the optimal simplex tableau and suppose the current constant coefficient is fractional; in this case would like to get a cutting plane; the question is what are the coefficients; these can be gotten from the cut generating function e.g. the "GMI" function

Some conditions that must be met for good cut-generating functions

Main Theorem on the Decomposition of Perturbation Space

If perturbation space is trivial then function π is "extreme"

Main theorem: under nice conditions for a minimal function π then some conditions on, perhaps, the cut-generating functions

2.3 On Compact Representations of Voronoi Cells of Lattices (presented by Christoph Hunkenschroder)

Lattice

Interested in a lattice; the integral linear span of a bunch of vectors

Another way to see it is at as a discrete subgroup of R^n under addition

Two bases that generate the same lattice differ only by a TU matrix

Define the dual lattice

Two Lattice Problems

Given a lattice what is the shortest nonzero vector in the lattice (SVP)

Given a lattice and a target vector find the vector in the lattice that is closest to this point (CVP)

Appear in: integer programming; cryptography

Known to be NP-hard

Some Algorithms

Kannan's enumerative algorithm for SVP and CVP with running time $n^{O(n)}$; enumerates all possible answers with polynomial time

Ajtai et al improving the running time to $2^{O(n)}$ but with randomness and exponential space

Micciancio et al. gave deterministic version (what this work is based on)

Fastest algorithms use $2^{n+o(n)}$ time with discrete Gaussian sampling but exonential space

Question: is the exponential space necessary for running time $2^{O(n)}$

Voronoi Cells of Lattice

The voronoi cell of a lattice is all points that are closest to 0;

Can be understood as the intersection of halfspaces; nicely not all halfspaces are needed; a centrally symmetric polytope with up to $2(2^n - 1)$ facets and up to $(n+1)!$ vertices

Suppose we had this lattice; vertex v is a closest point to t precisely when t is in the Voronoi cell of v; what Micciancio et al. do is follow line until find a facet, subtract it and repeat; exponential space is needed to store set of all facet vectors

c-Compact Basis

Imagine all facet vectors can be represented in a basis with small coefficients; then could iterate over all coefficients (end up with a superset of basis instead of basis itself but the original algorithm works fine with a superset); with such a basis could use just polynomial space

Upper bounds on Compactness Constant

This constant mentioned in Engel's work in '88

Seysen '99 gets an upper bound of $n^{O(\log n)}$

This work: gives an n^2 upper bound but also that there are lattices with $\Omega(n)$; also a class of lattices where the constant is always 1, namely zonotopes

Lemma: really just need to look for dual basis vectors so that the innter product with all facet vectors is small

2.4 A General Framework for Handling Commitment in Online Throughput Maximization (presented by Franziska Eberle)

Throughput Maximization

Given jobset with processing times, deadlines and release dates

Want to schedule all jobs so they complete on time

Scheduling is preemptive so can deschedule then reschedule a job

This problem is solvable via DP

In the considered talk these release dates are **online**

Performance Measures

In online optimization do **competitive analysis**: compare self to optimal offline optimum

Strong impossibility results: if a job is tight between release and deadline

Will therefore assume jobs have a lot of wiggle room given by parameter ϵ

Commitment Models

Give algorithms in 4 models with various compettitve ratios in 4 settings with functions of ϵ

1. Commitment upon arrival: right when a job arrives it learns if it will complete on time or not; in this setting no way of being competitive
2. δ-commitment: when slack becomes δ-small the scheduler has to commit to deleting job or not
3. Commitment upon admission: once job is started, scheduler commits to finishing on time
4. No commitment

This talk: one algorithmic framework for 2, 3, 4

Their Algorithm: Design Principles

1. Admission not too close to deadline ($\delta < \epsilon$)
2. Preemption only by "small" jobs (first two principles considered by previous work; but these alone cannot handle commitment)
3. Responsibility for preempted jobs; jobs aren't admitted until end of "region" of job that currently is running

Algorithm then consists of two parts: (1) and (2) scheduling

Regions

Job not scheduled until it is larger than the owner of the current region

Scheduling

Once jobs admitted, schedule in shortest processing time order

Observation: a job is always prioritized in its own region

Analysis

Two parts of analysis

1. Admission of enough jobs
2. Completion of enough admitted jobs

Preemption Tree

Idea used in proofs

Regions form a laminar structure captured by preemption tree.

If no job there, the machine job rules, once job is admitted its appended to machine job; children of jobs are those that preempt them

Part 1

Want to bound the nuber of jobs only scheduled by OPT in terms of the number of jobs the algorithm admits

Show that this holds on average throughout the whole preemption tree (for every preemption subtree)

Part 2

Lemma: all admitted jobs will complete on time

Lemma: at least half of all admitted jobs will complete on time in other setting

Outline

General algorithmic framework for commitment

Some impossibility results not given in talk

Future Work

Determine if commitment is actually harder or if weights are harder and extending results to m machines

2.5 The Markovian Price of Information (presented by Haotian Jiang)

How to Purchase a House

Some idea of the value of a house but don't have complete information of the value of houses

To really figure out the value of a house need to hire an inspector to evaluate the price

The utility, then, is the value of a house minus what we pay to learn

Question is how to max utility

Pandora's Box

$X_1, \ldots, X_n$ independent r.v. and given the distribution of each X_i and price π_i

Paying π_i fixes X_i at a draw of the random variable

Did out an example

MDP gives OPT strategy for Pandora's box

Unfortunately MDP has an exponential size state space

In '79 Weizman gave an efficient OPT strategy and then generalized by Dimitriu

Markov System

Markov chain $S = (V, P)$

Initial state s

Destination states $T = \{t_j\}$ where each corresponds to an outcome with reward $v_j \geq 0$

For each non destination state there is a proving price π_u; player can pay this to advance the current state; can collect reward at any "ready" state

A special case is Pandora's box

Goal of the player is to adaptively advance MSs to maxiize the expected reward obtained

Dimitriu showed an efficient optimal strategy but only worked for selecting \$$\leq$ 1\$MS

In this work interested in selecting $\leq k$ MS instead of ≤ 1 or even a matroid

Markovian Price of Information Model

Problem considered in this problem

In each step player can

1. pay price to advance non-ready MS
2. Select ready MS subject to packing constraints

Pandora's box is a special case as is combinatorial maximization (only 1 destination state)

Example with 2-uniform matroid

Theorem: for packing constraint $\mathcal{F}$ if there exists an α approximation greedy aglorithm this can be converted to an α approximation strategy for the MPOI model

E.g. get matroid: OPT; get matching 2-approx; etc.\

Question then is what is Their Transformation?

Grade

τ_u for state u in a Markov system is how "profitable" u is

Define a τ **penalized game** where all reward are increased by τ; as τ moves from $-\infty$ to ∞ there is some point where the transition where player wants to continue for 1 more step or give up; this is the **grade** of the state

Grade can be computed efficiently; is a variant of the Gittins index

Idea of Transformation

Want to use greedy algorithm to get an algorithm for MPOI

Need to create an instance to run "greedy"

Replace each S_u by a deterministic value; namely the grade

Transformation

Use grade as proxy and run "greedy"

Minimization Version

Can we get similar results for minimization problems?

Turns out can for a covering constraint instead of a packing constraint

Basic idea is to replace reward by costs: e.g. spanning tree, set cover and more results

Recap of MPOI Result

Defined MPOI model to extend multi-stage PB to handle packing constraints

Gave a generic transformation from a greedy algorithm that maintains the approximation factor

The transformation used grade as a proxy for states and ran greedy algorithm

2.6 An exact algorithm for robust influence maximization (presented by Emiliano Traversi)

Basic goal: study the spread of an agent in an environment, e.g.\ news in a social network; spread of fire; spread of disease etc

Main Ingrediants

1. Graph
2. Seeds: nodes where spread starts
3. Diffusion process
4. Activated: if it is affected by the diffusion process

Question is how, for a given number of seeds, can we maximize the total influence

Literature

Threshold model: node becomes active only if enough of its neighbors are activated more than some threshold

Determinstic / robust variants

Problem Formulation

As above but with a linear activation function: each edge has a weight; a node becomes active if sum of active nodes weighted by edges exceeds its threshold t_j

Can formulate problem as a bilevel LP: what nodes are active are one LP and which nodes are seeds is another LP

IMP with Activation Sets

A node is active iff there is some subset of nodes that are all active and meet the necesarry threshold constraints; a so called **activation set**

Can reformulate the previous LP as an LP over activation sets; problem is an exponential number of constraints in this reformulation

An advantage of this formulation is that all optimal solutions are integral

This is nice because can then dualize one LP and so the bilevel problem is a $\max\max$ problem and so really just a one level program; unfortunately a quadratic program

If certain variables are fixed then the LP obtained becomes integral

Then with some extra reasoning can turn quadratic problem into a MI(L)P

Robust IMP

Now suppose the thresholds of each node are uncertain

Can formulate this is a bilevel problem as before but now using activation sets doesn't make sense anymore; the threshold can be changed by the adversary

Basic idea: include dual variables for all the activation sets that may be chosen to be minimal by the adversary

3 May 24, 2019

3.1 Identically self-blocking clutters (presented by Ahmad Abdi)

Clutter is identacally self blocking if its blocker is itself

Theorem: every pair of blocking clutters has an identically self-blocking clutter

The Deltas

Look at degenerate projective plane

This is a clutter; also its blocker is itself

The Fano Plane

7 points, 7 lines, a projective plane

Nonideal also

Main Result

Every identically self-blocking clutter different from $\{\{a\}\}$ is nonideal

To prove it

- new lower bound on the packing number of an arbitrary clutter
- quadratic programming and **gauge duality**
- don't know where/what the fractional vertex is

Lower bounding the packing number

Let $\mu(C)$ be the max number of pairwise disjoint members of clutter C; show a lower bound of "α"

Comes from Motzkin + Straus: gives clique number of graph

Finding the packing number of a graph comes from finding the clique number

Get a formulat for packing number that is partially convex and partiall nonconvex will use Cauchy-Schwartz and Caratheodory on nonconvex part

Gauge Duality

Taking two blocking polyhedra and look at minimizing $x^T x$ in each polyhedra; the optimal values of values of these are reciprical

Will apply this to ideal clutters

Open Questions

How can we find the fractional vertex

How strong/weak is packing number lower bound

3.2 Integer Programming and Incidence Treedepth (presented by Dusan Knop)

In practice many parameters for NP hard problem; one such parameter is the largest constraint in your matrix

In parameterized complexity

XP: want to run in time $n^{f(k)}$ for some computable function f of your parameter FPT: get $f(k) \cdot n^c$

Gaifman Graphs

A simple ILP

Vertex for each variable; put an edge between two vertices if there is a constraint that they both occur in

In dual graph have vertex for each constraint; edge if that constraint shares variables

In incidence graph have a matching of variables and constraints

Classical Results

Lenstra + Kannan: Can solve an ILP in FPT time in the size of the aformentioned graphs

Papadimitriou + Eisenbrand and Weismantel: improved the running time with dependence on number of rows (soze of the graph above) and max constraint

But size of the graph is antiquated parameter to parameterize on

Treewidth

Most succesful parameter in graph algorithm design

Not really useful for integer programming though because ILP is hard even when the treewidth of the above graphs is small

Size of Vertex Cover

Metric between treewidth and size of the graph; but doesn't help; but combine with largest entry can get FPT

So now natural question is there something else inbetween

Treedepth

For low treedepth vertices there is a vertex which can be pulled out of graph to decrease tree depth by one

Any FPT algorithm if bounded largest entry and tree depth bounded for graphs above (other than incidence treedepth)

In this paper show that even if incidence tree depth is bounded ILP becomes NP hard

Proof Idea

Reduce from 3 SAT; use Chinese Remainder Theorem

3.3 Sparsity of integer solutions in the average case (presented by Joseph Paat)

Q: is there a sparse solution for an IP?

$\sigma(A, b)$ is the size of a minimal *feasible* solution

Goal: bound $\sigma(A, b)$ for different values of b

Gave an example graph that shows different sparsity values for different values of b; achieve a local minimum at $b = 17$ and then alternates 1 and 2 afterwards

Goal: bound $\sigma(A) := \max_b \sigma(A, b)$

A string of work done to bound $\sigma(A)$; Aliev bounds it in terms of $\det(AA^T)$ but this can't be efficienty computed so instead gave bound in terms of $||A||_\infty$

Q: is $\sigma(A)$ the "typical" value of $\sigma(A, b)$

A typical value in the previous example is 2 or 1 because if you u.a.r.\ sampled a b then you'd probably get a 1 or 2

In full journal version of paper show that $\Pr(\sigma(A, b)) \leq k$ for some values of b

Why Care

So why care about bounding sparsity for most bs or the above probabilities?

Main goal is to come up with a probabilistic algorithm to solve bilevel problems: want to maximize some function $f(x, y)$ where first you choose y and then x determined by program dependent on y

Main Result

Average value of sparsity bounded by $m + \max \Phi$ or $2m + \min \Phi$ where Φ depends on your matrix: Φ is the number of prime actors of $|\det(B)|$ where B is an $m \times m$ submatrix

Also can bound it in terms of $\Delta := |\det(B)|$

Thus, on average get pretty close to linear in m provided matrix A is not too nasty and that's the best you can hope for since in LPs need m

Provided there is at least two invertible submatrices then the average case bound is smaller than the max bound; because the average case determinant bound sums over all submatrices and the average case bounds they give is a max over submatrices

Prime Factors

Q: Why is $\sigma(A, b)$ affected by Φ?

Gave an example

Other Results

1. Worst case sparsity can be arbitrarily larger than average sparsity
2. One of their bounds holds for **optimal** solutions

3.4 Intersection Cuts for Polynomial Optimization (presented by Gonzalo Muñoz)

Outline

1. Intersection cuts
2. Intersections Cuts for polynomial optimization
3. Outer-Product-Free Sets
4. Numerical Experiments

Basic Cutting Plane

Given a solution x to an LP, how can we find a cutting plane? Can use **intersection cuts**

Introduced by Balas

Suppose solve an LP and get a fractional solution. Find a convex set not containing any integral points, find where convex set intersect polytope and cut through this

Main Ingredients

Need

- Convex set not containing integral points
- Polyhedron
- Integer points (or any closed set)

Formally will derive cuts using convex forbidden zone or S-free sets.

An S-free set C is maximal S-free if it is not contained in another S-free set

Polynomial Optimization

Minimize polynomial function f_0 subject to polynomial constraints

Use **moment-based reformulation**

Stack polynomial up to degree r; then can express polynomial as a matrix; transforms the polynomial constraint into a matrix inner product; end up with only linear expressions except for one nonlinear constraint

Outer-Product-Free-Sets

Start with eigenvalue decomposition of matrix; it's nearest outer product is $\lambda_1 \cdot v_1 v_1^T$

Thus if $\bar{X}$ is not outer product we can construct a ball which is outer product free; but this ball is normally small

Maximal OPF Sets are Cones

Conic closure of outerproduct free set is OPF

Theorem: half-space $< A, X >$ is maximal OPF iff A is negative semi-definite

Summary

Describe 5 families of cuts

1. Ball cut
2. Strengthened ball cut
3. Outer approximation
4. 2 x 2 cut
5. Genralized minor

Multiple ways of obtaining cutting planes via OPFs

Can be generated efficiently