CS 583: Approximation Algorithms	Lecture date: February 3, 2016
Instructor: Chandra Chekuri	Scribe: CC

Notes updated in Spring 2011 and Spring 2018.

In the previous lecture we discussed the KNAPSACK problem. In this lecture we discuss other *packing* and *independent set* problems. We first discuss an abstract model of packing problems. Let N be a finite ground set. A collection of $\mathcal{I} \subset 2^N$ of subsets of N is said to be *down closed* if the following property is true: $A \in \mathcal{I}$ implies that for all $B \subset A$, $B \in \mathcal{I}$. A down closed collection is also often called and *independence* system. The sets in $\mathcal{I}$ are called independent sets. Given an independence family $(N, \mathcal{I})$ and a non-negative weight function $w : N \to \mathbb{R}^+$ the maximum weight independent set problem is to find $\max_{S \in \mathcal{I}} w(S)$. That is, find an independent set in $\mathcal{I}$ of maximum weight. Often we may be interested in the setting where all weights are 1 in which case we wish to find the maximum cardinality independent set. We discuss some canonical examples.

Example: Independent sets in graphs: Given a graph $G = (V, E)$
$\mathcal{I} = \{S \subseteq V \mid \text{there are no edges between nodes in } S\}$. Here the ground set is V. There are many interesting special cases of the graph problem. For instance problems arising from geometric objects such as intervals, rectangles, disks and others.

Example: Matchings in graphs: Given a graph $G = (V, E)$ let $\mathcal{I} = \{M \subseteq E \mid M \text{ is a matching in } G\}$. Here the ground set is E.

Example: Matroids: A matroid $\mathcal{M} = (N, \mathcal{I})$ is defined as a system where $\mathcal{I}$ is down closed and in addition satisfies the following key property: if $A, B \in \mathcal{I}$ and $|B| > |A|$ then there is an element $e \in B \setminus A$ such that $A \cup \{e\} \in \mathcal{I}$. There are many examples of matroids. We will not go into details here.

Example: Intersections of independence systems: given some k independence systems on the same ground set $(N, \mathcal{I}_1), (N, \mathcal{I}_2), \ldots, (N, \mathcal{I}_k)$ the system defined by $(N, \mathcal{I}_1 \cap \mathcal{I}_2 \ldots \cap \mathcal{I}_k)$ is also an independence system. Well-known examples include intersections of matroids.

1 Maximum Independent Set Problem in Graphs

A basic graph optimization problem with many applications is the *maximum (weighted) independent set problem* (MIS) in graphs.

Definition 1 *Given an undirected graph $G = (V, E)$ a subset of nodes $S \subseteq V$ is an independent set (stable set) iff there is no edge in E between any two nodes in S. A subset of nodes S is a* clique *if every pair of nodes in S have an edge between them in G.*

The MIS problem is the following: given a graph $G = (V, E)$ find an independent set in G of maximum cardinality. In the weighted case, each node $v \in V$ has an associated non-negative weight $w(v)$ and the goal is to find a maximum weight independent set. This problem is NP-Hard

and it is natural to ask for approximation algorithms. Unfortunately, as the famous theorem below shows, the problem is extremely hard to approximate.

Theorem 1 (Håstad [3]) *Unless $P = NP$ there is no $\frac{1}{n^{1-\epsilon}}$-approximation for MIS for any fixed $\epsilon > 0$ where n is the number of nodes in the given graph.*

Remark: The maximum clique problem is to find the maximum cardinality clique in a given graph. It is approximation-equivalent to the MIS problem; simply complement the graph.

The theorem basically says the following: there are a class of graphs in which the maximum independent set size is either less than n^{δ} or greater than $n^{1-\delta}$ and it is NP-Complete to decide whether a given graph falls into the former category or the latter.

The lower bound result suggests that one should focus on special cases, and several interesting positive results are known. First, we consider a simple greedy algorithm for the unweighted problem.

```
GREEDY(G):
S ← ∅
While G is not empty do
   Let v be a node of minimum degree in G
   S ← S ∪ {v}
   Remove v and its neighbors from G
end while
Output S
```

Theorem 2 *Greedy outputs an independent set S such that $|S| \geq n/(\Delta + 1)$ where Δ is the maximum degree of any node in the graph. Moreover $|S| \geq \alpha(G)/\Delta$ where $\alpha(G)$ is the cardinality of the largest independent set. Thus Greedy is a $1/\Delta$ approximation.*

Proof: We upper bound the number of nodes in $V \setminus S$ as follows. A node u is in $V \setminus S$ because it is removed as a neighbor of some node $v \in S$ when Greedy added v to S. Charge u to v. A node $v \in S$ can be charged at most Δ times since it has at most Δ neighbors. Hence we have that $|V \setminus S| \leq \Delta|S|$. Since every node is either in S or $V \setminus S$ we have $|S| + |V \setminus S| = n$ and therefore $(\Delta + 1)|S| \geq n$ which implies that $|S| \geq n/(\Delta + 1)$.

We now argue that $|S| \geq \alpha(G)/\Delta$. Let S^* be a largest independent set in G. As in the above proof we can charge each node v in $S^* \setminus S$ to a node $u \in S \setminus S^*$ which is a neighbor of v. The number of nodes charged to a node $u \in S \setminus S^*$ is at most Δ. Thus $|S^* \setminus S| \leq \Delta|S \setminus S^*|$.

□

Exercise: Show that Greedy outputs an independent set of size at least $\frac{n}{2(d+1)}$ where d is the *average* degree of G.

Remark: The well-known Turan's theorem shows via a clever argument that there is always an independent set of size $\frac{n}{(d+1)}$ where d is the average degree of G.

Remark: For the case of unweighted graphs one can obtain an approximation ratio of $\Omega(\frac{\log d}{d \log \log d})$ where d is the average degree. Surprisingly, under a complexity theory conjecture called the Unique-Games conjecture it is known to be NP-Hard to approximate MIS to within a factor of $O(\frac{\log^2 \Delta}{\Delta})$ in graphs with maximum degree Δ when Δ is sufficiently large.

Exercise: Consider the weigthed MIS problem on graphs of maximum degree Δ. Alter Greedy to sort the nodes in non-increasing order of the weight and show that it gives a $\frac{1}{\Delta}$-approximation. Can one obtain an $\Omega(1/d)$-approximation for the weighted case where d is the average degree?

LP Relaxation: One can formulate a simple linear-programming relaxation for the (weighted) MIS problem where we have a variable $x(v)$ for each node $v \in V$ indicating whether v is chosen in the independent set or not. We have constraints which state that for each edge (u, v) only one of u or v can be chosen.

$$\begin{aligned} \text{maximize } & \sum_{v \in V} w(v)x(v) \\ \text{subject to } & x(u) + x(v) \leq 1 \qquad (u,v) \in E \\ & x(v) \in [0,1] \qquad v \in V \end{aligned}$$

Although the above is a valid integer programming relaxation of MIS when the variabels are constrained to be in $\{0, 1\}$, it is not a particularly useful formulation for the following simple reason.

Claim 3 *For* any *graph the optimum value of the above LP relaxation is at least $w(V)/2$. In particular, for the unweighted case it is at least $n/2$.*

Simply set each $x(v)$ to 1/2!

One can obtain a *strengthened* formulation below by observing that if S is clique in G then any independent set can pick at most one node from S.

$$\begin{aligned} \text{maximize } & \sum_{v \in V} w(v)x(v) \\ \text{subject to } & \sum_{v \in S} x(v) \leq 1 \qquad S \text{ is a clique in } G \\ & x(v) \in [0,1] \qquad v \in V \end{aligned}$$

The above linear program has an exponential number of constraints, and it cannot be solved in polynomial time in general, but for some special cases of interest the above linear program can indeed be solved (or approximately solved) in polynomial time and leads to either exact algorithms or good approximation bounds.

Approximability of Vertex Cover **and** MIS**:** The following is a basic fact and is easy to prove.

Proposition 4 *In any graph $G = (V, E)$, S is a vertex cover in G if and only if $V \setminus S$ is an independent set in G. Thus $\alpha(G) + \beta(G) = |V|$ where $\alpha(G)$ is the size of a maximum independent set in G and $\beta(G)$ is the size of a minimum vertex cover in G.*

The above shows that if one of Vertex Cover or MIS is NP-Hard then the other is as well. We have seen that Vertex Cover admits a 2-approximation while MIS admits no constant factor approximation. It is useful to see why a 2-approximation for Vertex Cover does not give any

useful information for MIS even though $\alpha(G) + \beta(G) = |V|$. Suppose S^* is an optimal vertex cover and has size $\geq |V|/2$. Then a 2-approximation algorithm is only guaranteed to give a vertex cover of size $|V|$! Hence one does not obtain a non-trivial independent set by complementing the approximate vertex cover.

Some special cases of MIS**:** We mention some special cases of MIS that have been considered in the literature, this is by no means an exhaustive list.

- Interval graphs; these are intersection graphs of intervals on a line. An exact algorithm can be obtained via dynamic programming and one can solve more general versions via linear programming methods.
- Note that a maximum (weight) matching in a graph G can be viewed as a maximum (weight) independent set in the line-graph of G and can be solved exactly in polynomial time. This has been extended to what are known as claw-free graphs.
- Planar graphs and generalizations to bounded-genus graphs, and graphs that exclude a fixed minor. For such graphs one can obtain a PTAS due to ideas originally from Brenda Baker.
- Geometric intersection graphs. For example, given n disks on the plane find a maximum number of disks that do not overlap. One could consider other (convex) shapes such as axis parallel rectangles, line segments, pseudo-disks etc. A number of results are known. For example a PTAS is known for disks in the plane. An $\Omega(\frac{1}{\log n})$-approximation for axis-parallel rectangles in the plane when the rectangles are weighted and an $\Omega(\frac{1}{\log \log n})$-approximation for the unweighted case.

1.1 Elimination Orders and MIS

We have seen that a simple Greedy algorithm gives a Δ-approximation for MIS in graphs with max degree Δ. One can also get a Δ approximation for a larger class of Δ-degenerate graphs. To motivate degenerate graphs consider the class of planar graphs. The maximum degree of a planar graph need not be small. Nevertheless, via Euler's theorem, we know that every planar graph has a vertex of degree at most 5 since the maximum number of edges in a planar graph is at most $3n - 6$. Moreover, every subgraph of a planar graph is planar, and hence the Greedy algorithm will repeatedly find a vertex of degree at most 5 in each iteration. From this one can show that Greedy gives a 1/5-approximation for MIS in planar graphs. Now consider the intersection graph of a collection of intervals on the real line. That is, we are given n intervals $I_1, I_2, \ldots, I_n$ where each $I_i = [a_i, b_i]$ for real numbers $a_i \leq b_i$. The goal is to find a maximum number of the intervals in the given set of intervals which do not overlap. This is the same as finding MIS in the *intersection graph* of the intervals - the graph is obtained by creating a vertex v_i for each I_i, and by adding edges $v_i v_j$ if I_i and I_j overlap. It is well-known that greedily picking intervals in earliest finish time order (ordering them according to b_i values) is optimal; the reader should try to prove this. Can one understand the analysis of all these examples in a unified fashion? Yes. For this purpose we consider the class of inductive k-independent graphs considered by by Akcoglu et al. [1] and later again by Ye and Borodin [7].

For a vertex v in a graph we use $N(v)$ denote the neighbors of v (not including v itself). For a graph $G = (V, E)$ and $S \subset V$ we use $G[S]$ to denote the subgraph of G induced by S.

Definition 2 *An undirected graph $G = (V, E)$ is inductive k-independent if there is an ordering of the vertices $v_1, v_2, \ldots, v_n$ such that for $1 \leq i \leq n$, $\alpha(G[N(v_i) \cap \{v_{i+1}, \ldots, v_n\}]) \leq k$.*

Graphs which are inductively 1-independent have a *perfect* elimination ordering and are called chordal graphs because they have an alternate characterization. A graph is chordal iff each cycle C in G has a chord (an edge connecting two nodes of C which is not an edge of C), or in other words there is no induced cycle of length more than 3.

Exercise: Prove that the intersection graph of intervals is chordal.

Exercise: Prove that if $\Delta(G) \leq k$ then G is inductively k-independent. Prove that if G is k-degenerate then G is inductively k-independent.

The preceding shows that planar graphs are inductively 5-independent. In fact one can show some thing stronger. They are inductively 3-independent. Given a graph G one can ask whether there is an algorithm that checks whether G is inductively k-independent. There is such an algorithm that runs in time $O(k^2 n^{k+2})$ [7]. A classical result shows how to recognize chordal graphs ($k = 1$) in linear time. However, most of the useful applications arise by showing that a certain class of graphs are inductively k-independent for some small value of k. See [7] for several examples.

Exercise: Prove that the Greedy algorithm that considers the vertices in the inductive k-independent order gives a $\frac{1}{k}$-approximation for MIS.

Interestingly one can obtain a $\frac{1}{k}$-approximation for the maximum weight independent set problem in inductively k-independent graphs. The algorithm is simple and runs in linear time but is not obvious. To see this consider the weighted problem for intervals. The standard algorithm to solve this is via dynamic programming. However, one can obtain an optimum solution for all chordal graphs (given the ordering). We refer the reader to [7] for the algorithm and proof (originally from [1]). Showing a $\Omega(1/k)$-approximation is easier.

2 The efficacy of the Greedy algorithm for a class of Independence Families

The Greedy algorithm can be defined easily for an arbitrary independence system. It iteratively adds the best element to the current independent set while maintaining feasibility. Note that the implementation of the algorithm requires having an oracle to find the best element to add to a current independent set S.

```
GREEDY((N, I)):
S ← ∅
While (TRUE)
  Let A ← {e ∈ N \ S | S + e ∈ I}
  If A = ∅ break
  e ← argmax_{e∈A} w(e)
  S ← S ∪ {e}
endWhile
Output S
```

Exercise: Prove that the Greedy algorithm gives a 1/2-approximation for the maximum weight matching problem in a general graph. Also prove that this bound is tight even in bipartite graphs. Note that max weight matching can be solved exactly in polynomial time.

Remark: It is well-known that the Greedy algorithm gives an optimum solution when $(N, \mathcal{I})$ is a matroid. Kruskal's algorithm for min/max weight spanning tree is a special case of this fact.

It is easy to see that Greedy does poorly for MIS problem in general graphs. A natural question is what properties of $\mathcal{I}$ enable some reasonable performance guarantee for Greedy. A very general result in this context has been established due to Jenkyn's generalizing several previous results. In order to state the result we set up some notation. Given an independence system $(N, \mathcal{I})$ we say that a set $A \in \mathcal{I}$ is a *base* if it is a *maximal* independent set. It is well-known that in a matroid $\mathcal{M}$ all bases have the same cardinality. However this is not true in general independence system.

Definition 3 *An independence system $(N, \mathcal{I})$ is a k-system if for any two bases $A, B \in \mathcal{I}$, $|A| \leq k|B|$. That is, the ratio of the cardinality of a maximum base and the cardinality of a minimum base is at most k.*

The following theorem is not too difficult but not so obvious either.

Theorem 5 *Greedy gives a $1/k$-approximation for the maximum weight independent set problem in a k-system.*

The above theorem generalizes and unifies several examples that we have seen so far including MIS in bounded degree graphs, matchings, matroids etc. How does one see that a given independence system is indeed a k-system for some parameter k? For instance matchings in graphs form a 2-system. The following simple lemma gives an easy way to argue that a given system is a k-system.

Lemma 6 *Suppose $(N, \mathcal{I})$ is an independence system with the following property: for any $A \in \mathcal{I}$ and $e \in N \setminus A$ there is a set of at most $Y \subset A$ such that $|Y| \leq k$ and $(A \setminus Y) \cup \{e\} \in \mathcal{I}$. Then $\mathcal{I}$ is a k-system.*

We leave the proof of the above as an exercise.

We refer the reader to [5, 6] for analysis of Greedy in k-systems and other special cases.

3 Randomized Rounding with Alteration for Packing Problems

The purpose of this section to highlight a technique for rounding LP relaxations for packing problems. We will consider a simple example, namely the maximum weight independent set problem in interval graphs. Recall that we are given n intervals $I_1, I_2, \ldots, I_n$ with non-negative weights $w_1, \ldots, w_n$ and the goal is to find a maximum weight subset of them which do not overlap. Let $I_i = [a_i, b_i]$ and let $p_1, p_2, \ldots, p_m$ be the collection of end points of the intervals. We can write a simple LP relaxation for this problem. For each interval i we have a variable $x_i \in [0, 1]$ to indicate whether I_i is chosen or not. For each point p_j, among all intervals that contain it, at most one can be chosen. These are clique constraints in the underlying interval graph.

$$\begin{aligned}
\text{maximize } & \sum_{i=1}^{n} w_i x_i \\
\text{subject to } & \sum_{i: p_j \in I_i} x_i \leq 1 \qquad 1 \leq j \leq m \\
& x_i \in [0,1] \qquad 1 \leq i \leq n
\end{aligned}$$

Note that it is important to retain the constraint that $x_i \leq 1$. Interestingly it is known that the LP relaxation defines an integer polytope and hence one can solve the integer program by solving the LP relaxation! This is because the incidence matrix defining the LP is totally unimodular (TUM). We refer the reader to books on combinatorial optimization for further background on this topic. Here we assume that we do not know the integer properties of the LP. We will round it via a technique that is powerful and generalizes to NP-Hard variants of the interval scheduling problem among many others.

Suppose we solve the LP and obtain an optimum fraction solution x^*. We have $\sum_i w_i x_i^* \geq OPT$. How do we round to obtain an integer solution whose value is close to that of OPT? Suppose we randomly choose I_i with probablility cx_i^* for some $c \leq 1$. Let R be the random set of chosen intervals. Then the expected weight of R, by linearity of expectation, is $c\sum_i w_i x_i^* \geq c \cdot OPT$. However, it is highly likely that the random solution R is not going to be feasible. Some constraint will be violated. The question is how we can *fix* or *alter* R to find a subset $R' \subseteq R$ such that R' is a feasible solution *and* the expected value of R' is not too much smaller than that of R. This depends on the independence structure.

Here we illustrate this via the interval problem. Without loss of generality we assume that $I_1, \ldots, I_n$ are sorted by their right end point. In other words the order is a perfect elimination order for the underlying interval graph.

ROUNDING-WITH-ALTERATION:
Let x be an optimum fractional solution
Round each i to 1 independently with probability $x_i/2$. Let x' be rounded solution.
$R \leftarrow \{i \mid x'_i = 1\}$
$S \leftarrow \emptyset$
For $i = n$ **down to** 1 do
  If $(i \in R)$ *and* $(S \cup \{i\}$ is feasible)
    $S \leftarrow S \cup \{i\}$
  EndIf
Endfor
Output feasible solution S

The algorithm consists of two phases. The first phase is a simple selection phase via independent randomized rounding. The second phase is deterministic and is a greedy pruning step in the *reverse* elimination order. To analyze the expected value of S we consider two binary random variables for each i, Y_i and Z_i. Y_i is 1 if $i \in R$ and 0 otherwise. Z_i is 1 if $i \in S$ and 0 otherwise.

By linearity of expectation,

Claim 7 $\mathbb{E}[w(S)] = \sum_i w_i \mathbb{E}[Z_i] = \sum_i w_i \Pr[Z_i = 1]$.

Via the independent randomized rounding in the algorithm.

Claim 8 $\Pr[Y_i = 1] = x_i/2$.

How do we analyze $\Pr[Z_i = 1]$? The random variables $Z_1, \ldots, Z_n$ are not independent and could be highly correlated even though $Y_1, \ldots, Y_n$ are independent. For this purpose we try to understand $\Pr[Z_i = 0 \mid Y_i = 1]$ which is the conditional probability that an interval I_i that is chosen in the first step is rejected in the pruning phase. We often would not be able to get an exact estimate of this quantity but we can upper bound it as follows. Here the ordering plays a crucial role. Why would I_i be rejected in the pruning phase? Note that when I_i is considered in the pruning phase, the only intervals that have been considered have their right end points after the right end point of I_i. Let $A_i = \{j \mid j > i \text{ and } I_j \text{ and } I_i \text{ intersect at } b_i\}$ be the potential set of intervals that can cause i to be rejected. Recall that the LP implies the following constraint:

$$x_i + \sum_{j \in A} x_j \leq 1$$

at the point b_j. Let $\mathcal{E}_1$ be the event that I_i is rejected in the pruning phase. Let $\mathcal{E}_\in$ be the event that at least one of the intervals in A is selected in the *first phase*. Note that $\mathcal{E}_1$ can happen only if $\mathcal{E}_2$ happens. Thus $\Pr[\mathcal{E}_1] \leq \Pr[\mathcal{E}_2]$. In general we try to upper bound $\Pr[\mathcal{E}_2]$. In this simple case we have an exact formula for it.

$$\Pr[\mathcal{E}_2] = 1 - \prod_{j \in A} \Pr[Y_j = 0] = 1 - \prod_{j \in A} (1 - x_j/2).$$

We claim that $\Pr[\mathcal{E}_2] \leq \sum_{j \in A} x_j/2 \leq 1/2$. One can derive this by showing that $\prod_{j \in A}(1 - x_j/2)$ subject to $\sum_{j \in A} x_j/2 \leq 1/2$ is at least $1/2$. Another way of doing this is via Markov's inequality. Let $T = \sum_{j \in A} Y_j$ be the number of intervals from A selected in the first phase. $E[T] \leq \sum_{j \in A} x_j/2 < 1/2$. By Markov's inequality $\Pr[T \geq 2E[T]] \leq 1/2$. $\mathcal{E}_2$ is the event that $\Pr[T \geq 1]$.

Using the claim,

$$\Pr[Z_i = 1 \mid Y_i = 1] = 1 - \Pr[Z_i = 0 | Y_i = 1] \geq 1/2.$$

This allows us to lower bound the expected weight of the solution output by the algorithm, and yields a randomized $1/4$ approximation.

Claim 9 $\mathbb{E}[w(S)] \geq \sum_i w_i x_i/4$.

Proof: We have

$$\mathbb{E}[w(S)] = \sum_i w_i \Pr[Z_i = 1] = \sum_i w_i \Pr[Y_i = 1] \Pr[Z_i = 1 \mid Y_i = 1] \geq \sum_i w_i (\frac{x_i}{4} \cdot \frac{1}{2}) \geq \sum_i w_i x_i/4.$$

□

This type of rounding has applications to a variety of settings - see [2] for applications and the general framework called *contention resolution schemes*.

4 Packing Integer Programs (PIPs)

We can express the KNAPSACK problem as the following integer program. We scaled the knapsack capacity to 1 without loss of generality.

$$\begin{aligned} \text{maximize} \quad & \sum_{i=1}^{n} p_i x_i \\ \text{subject to} \quad & \sum_{i} s_i x_i \leq 1 \\ & x_i \in \{0,1\} \qquad 1 \leq i \leq n \end{aligned}$$

More generally if have multiple linear constraints on the "items" we obtain the following integer program.

Definition 4 *A packing integer program (PIP) is an integer program of the form* $\max\{wx \mid Ax \leq 1, x \in \{0,1\}^n\}$ *where* w *is a* $1 \times n$ *non-negative vector and* A *is a* $m \times n$ *matrix with entries in* $[0,1]$. *We call it a* $\{0,1\}$*-PIP if all entries are in* $\{0,1\}$.

In some cases it is useful/natural to define the problem as $\max\{wx \mid Ax \leq b, x \in \{0,1\}^n\}$ where entries in A and b are required to rational/integer valued. We can convert it into the above form by dividing each row of A by b_i.

When m the number of rows of A (equivalently the constraints) is small the problem is tractable. It is some times called the m-dimensional knapsack and one can obtain a PTAS for any fixed constant m. However, when m is large we observe that MIS can be cast as a special case of $\{0,1\}$-PIP. It corresponds exactly to the simple integer/linear program that we saw in the previous section. Therefore the problem is at least as hard to approximate as MIS. Here we show via a clever LP-rounding idea that one can generalize the notion of bounded-degree to *column-sparsity* in PIPs and obtain a related approximation. We will then introduce the notion of *width* of the constraints and show how it allows for improved bounds.

Definition 5 *A PIP is* k*-column-sparse if the number of non-zero entries in each column of* A *is at most* k. *A PIP has width* W *if* $\max_{i,j} A_{ij}/b_i \leq 1/W$.

4.1 Randomized Rounding with Alteration for PIPs

We saw that randomized rounding gave an $O(\log n)$ approximation algorithm for the SET COVER problem which is a canonical covering problem. Here we will consider the use of randomized rounding for packing problems. Let x be an optimum fractional solution to the natural LP relaxation of a PIP where we replace the constraint $x \in \{0,1\}^n$ by $x \in [0,1]^n$. Suppose we apply independent randomized rounding where we set x'_i to 1 with probability x_i. Let x' be the resulting integer solution. The expected weight of this solution is exactly $\sum_i w_i x_i$ which is the LP solution value. However, x' may not satisfy the constraints given by $Ax \leq b$. A natural strategy to try to satisfy the constraints is to set x'_1 to 1 with probability cx_i where $c < 1$ is some scaling constant. This may help in satisfying the constraints because the scaling creates some room in the constraints; we now have that the expected solution value is $c \sum_i w_i x_i$, a loss of a factor of c. Scaling by itself does not

allow us to claim that all constraints are satisfied with good probability. A very useful technique in this context is the technique of *alteration*; we judiciously fix/alter the rounded solution x' to force it to satisfy the constraints by setting some of the variables that are 1 in x' to 0. The trick is to do this in such a way as to have a handle on the final probability that a variable is set to 1. We will illustrate this for the KNAPSACK problem and then generalize the idea to k-sparse PIPs. The algorithms we present are from [4]. See [2] for further applications and related problems.

Rounding for Knapsack: Consider the KNAPSACK problem. It is convenient to think of this in the context of PIPs. So we have $ax \leq 1$ where a_i now represents the size of item i and the knapsack capacity is 1; w_i is the weight of item. Suppose x is a fractional solution. Call an item i "big" if $a_i > 1/2$ and otherwise it is "small". Let S be the indices of small items and B the indices of the big items. Consider the following rounding algorithm.

ROUNDING-WITH-ALTERATION FOR KNAPSACK:
Let x be an optimum fractional solution
Round each i to 1 independently with probability $x_i/4$. Let x' be rounded solution.
$x'' = x'$
If ($x'_i = 1$ for exactly one big item i)
  For each $j \neq i$ set $x''_j = 0$
Else If ($\sum_{i \in S} s_i x'_i > 1$ *or* two or more big items are chosen in x')
  For each j set $x''_j = 0$
End If
Output feasible solution x''

In words, the algorithm alters the rounded solution x' as follows. If exactly one big item is chosen in x' then the algorithm retains that item and rejects all the other small items. Otherwise, the algorithm rejects all items if two or more big items are chosen in x' or if the total size of all small items chosen in x' exceeds the capacity.

The following claim is easy to verify.

Claim 10 *The integer solution x'' is feasible.*

Now let us analyze the probability of an item i being present in the final solution. Let $\mathcal{E}_1$ be the event that $\sum_{i \in S} a_i x'_i > 1$, that is the sum of the sizes of the small items chose in x' exceeds the capacity. Let $\mathcal{E}_2$ be the event that at least one big item is chosen in x'.

Claim 11 $\Pr[\mathcal{E}_1] \leq 1/4$.

Proof: Let $X_s = \sum_{i \in S} a_i x'_i$ be the random variable that measures the sum of the sizes of the small items chosen. We have, by linearity of expectation, that

$$\mathbb{E}[X_s] = \sum_{i \in S} a_i \mathbb{E}[x'_i] = \sum_{i \in S} a_i x_i/4 \leq 1/4.$$

By Markov's inequality, $\Pr[X_s > 1] \leq \mathbb{E}[X_s]/1 \leq 1/4$. □

Claim 12 $\Pr[\mathcal{E}_2] \leq 1/2$.

Proof: Since the size of each big item in B is at least $1/2$, we have $1 \geq \sum_{i \in B} a_i x_i \geq \sum_{i \in B} x_i/2$. Therefore $\sum_{i \in B} x_i/4 \leq 1/2$. Event $\mathcal{E}_2$ happens if some item $i \in B$ is chosen in the random selection. Since i is chosen with probability $x_i/4$, by the union bound, $\Pr[\mathcal{E}_2] \leq \sum_{i \in B} x_i/4 \leq 1/2$. □

Lemma 13 *Let Z_i be the indicator random variable that is 1 if $x''_i = 1$ and 0 otherwise. Then $\mathbb{E}[Z_i] = \Pr[Z_i = 1] \geq x_i/16$.*

Proof: We consider the binary random variable X_i which is 1 if $x'_i = 1$. We have $\mathbb{E}[X_i] = \Pr[X_i = 1] = x_i/4$. We write

$$\Pr[Z_i = 1] = \Pr[X_i = 1] \cdot \Pr[Z_i = 1 \mid X_i = 1] = \frac{x_i}{4} \Pr[Z_i = 1 \mid X_i = 1].$$

To lower bound $\Pr[Z_i = 1 \mid X_i = 1]$ we upper bound the probability $\Pr[Z_i = 0 | X_i = 1]$, that is, the probability that we reject i conditioned on the fact that it is chosen in the random solution x'.

First consider a big item i that is chosen in x'. Then i is rejected iff if *another* big item is chosen in x'; the probability of this can be upper bounded by $\Pr[\mathcal{E}_1]$. If item i is small then it is rejected if and only if $\mathcal{E}_2$ happens or if a big item is chosen which happens with $\Pr[\mathcal{E}_1]$. In either case

$$\Pr[Z_i = 0 | X_i = 1] \leq \Pr[\mathcal{E}_1] + \Pr[\mathcal{E}_2] \leq 1/4 + 1/2 = 3/4.$$

Thus,

$$\Pr[Z_i = 1] = \Pr[X_i = 1] \cdot \Pr[Z_i = 1 \mid X_i = 1] = \frac{x_i}{4}(1 - \Pr[Z_i = 0 \mid X_i = 1]) \geq \frac{x_i}{16}.$$

□

One can improve the above analysis to show that $\Pr[Z_i = 1] \geq x_i/8$.

Theorem 14 *The randomized algorithm outputs a feasible solution of expected weight at least $\sum_{i=1}^{n} w_i x_i/16$.*

Proof: The expected weight of the output is

$$\mathbb{E}[\sum_i w_i x''_i] = \sum_i w_i \mathbb{E}[Z_i] \geq \sum_i w_i x_i/16$$

where we used the previous lemma to lower bound $\mathbb{E}[Z_i]$. □

Rounding for k-sparse PIPs: We now extend the rounding algorithm and analysis above to k-sparse PIPs. Let x be a feasible fractional solution to $\max\{wx \mid Ax \leq 1, x \in [0,1]^n\}$. For a column index i we let $N(i) = \{j \mid A_{j,i} > 0\}$ be the indices of the rows in which i has a non-zero entry. Since A is k-column-sparse we have that $|N(i)| \leq k$ for $1 \leq i \leq n$. When we have more than one constraint we cannot classify an item/index i as big or small since it may be big for some constraints and small for others. We say that i is small for constraint $j \in N(i)$ if $A_{j,i} \leq 1/2$ otherwise i is big for constraint j. Let $S_j = \{i \mid j \in N(i), \text{ and } i \text{ small for } j\}$ be the set of all small columns for j and $B_j = \{i \mid j \in N(i), \text{ and } i \text{ small for } j\}$ be the set of all big columns for j. Note that $S_j \cap B_j$ is the set of all i with $A_{j,i} > 0$.

ROUNDING-WITH-ALTERATION FOR k-SPARSE PIPS:
Let x be an optimum fractional solution
Round each i to 1 independently with probability $x_i/(4k)$. Let x' be rounded solution.
$x'' = x'$
For $j = 1$ to m do
  If ($x'_i = 1$ for exactly one $i \in B_j$)
    For each $h \in S_j \cup B_j$ and $h \neq i$ set $x''_h = 0$
  Else If ($\sum_{i \in S_j} A_{j,i} x'_i > 1$ *or* two or more items from B_j are chosen in x')
    For each $h \in S_j \cup B_j$ set $x''_h = 0$
  End If
End For
Output feasible solution x''

The algorithm, after picking the random solution x', alters it as follows: it applies the previous algorithm's strategy to each constraint j separately. Thus an element i can be rejected at different constraints $j \in N(i)$. We need to bound the total probability of rejection. As before, the following claim is easy to verify.

Claim 15 *The integer solution x'' is feasible.*

Now let us analyze the probability of an item i being present in the final solution. Let $\mathcal{E}_1(j)$ be the event that $\sum_{i \in S_j} A_{j,i} x'_i > 1$, that is the sum of the sizes of the items that are small for j in x' exceed the capacity. Let $\mathcal{E}_2(j)$ be the event that at least one big item for j is chosen in x'. The following claims follow from the same reasoning as the ones before with the only change being the scaling factor.

Claim 16 $\Pr[\mathcal{E}_1(j)] \leq 1/(4k)$.

Claim 17 $\Pr[\mathcal{E}_2(j)] \leq 1/(2k)$.

Lemma 18 *Let Z_i be the indicator random variable that is 1 if $x''_i = 1$ and 0 otherwise. Then $\mathbb{E}[Z_i] = \Pr[Z_i = 1] \geq x_i/(16k)$.*

Proof: We consider the binary random variable X_i which is 1 if $x'_i = 1$ after the randomized rounding. We have $\mathbb{E}[X_i] = \Pr[X_i = 1] = x_i/(4k)$. We write

$$\Pr[Z_i = 1] = \Pr[X_i = 1] \cdot \Pr[Z_i = 1 \mid X_i = 1] = \frac{x_i}{4k} \Pr[Z_i = 1 \mid X_i = 1].$$

We upper bound the probability $\Pr[Z_i = 0 | X_i = 1]$, that is, the probability that we reject i conditioned on the fact that it is chosen in the random solution x'. We observe that

$$\Pr[Z_i = 0 | X_i = 1] \leq \sum_{j \in N(i)} (\Pr[\mathcal{E}_1(j)] + \Pr[\mathcal{E}_2(j)] \leq k(1/(4k) + 1/(2k)) \leq 3/4.$$

We used the fact that $N(i) \leq k$ and the claims above. Therefore,

$$\Pr[Z_i = 1] = \Pr[X_i = 1] \cdot \Pr[Z_i = 1 \mid X_i = 1] = \frac{x_i}{4k}(1 - \Pr[Z_i = 0 \mid X_i = 1]) \geq \frac{x_i}{16k}.$$

□

The theorem below follows by using the above lemma and linearity of expectation to compare the expected weight of the output of the randomized algorithm with that of the fractional solution.

Theorem 19 *The randomized algorithm outputs a feasible solution of expected weight at least* $\sum_{i=1}^{n} w_i x_i/(16k)$. *There is* $1/(16k)$*-approximation for* k*-sparse PIPs.*

Larger width helps: We saw during the discussion on the KNAPSACK problem that if all items are small with respect to the capacity constraint then one can obtain better approximations. For PIPs we defined the *width* of a given instance as W if $\max_{i,j} A_{ij}/b_i \leq 1/W$; in other words no single item is more than $1/W$ times the capacity of any constraint. One can show using a very similar algorithm and anaylisis as above that the approximation bound improves to $\Omega(1/k^{\lceil W \rceil})$ for instance with width W. Thus if $W = 2$ we get a $\Omega(1/\sqrt{k})$ approximation instead of $\Omega(1/k)$-approximation. More generally when $W \geq c \log k/\epsilon$ for some sufficiently large constant c we can get a $(1-\epsilon)$-approximation. Thus, in the setting with multiple knapsack constraints, the notion of small with respect to capacities is that in each constraint the size of the item is $\leq \frac{c\epsilon}{\log k}$ times the capacity of that constraint.

References

[1] Karhan Akcoglu, James Aspnes, Bhaskar DasGupta, and Ming-Yang Kao. Opportunity cost algorithms for combinatorial auctions. In *Computational Methods in Decision-Making, Economics and Finance*, pages 455–479. Springer, 2002.

[2] C. Chekuri, J. Vondrák and R. Zenklusen. Submodular Function Maximization via the Multilinear Relaxation and Contention Resolution Schemes *SIAM J. on Computing*, 43(6): 1831–1879, 2014.

[3] J. Håstad. Clique is Hard to Approximate within $n^{1-\epsilon}$. *Acta Mathematica*, 182:105–142, 1999.

[4] N. Bansal, N. Korula, V. Nagarajan, A. Srinivasan. On k-Column Sparse Packing Programs. *Proc. of IPCO*, 2010. Available at `http://arxiv.org/abs/0908.2256`.

[5] Julian Mestre. Greedy in Approximation Algorithms. *Proc. of ESA*, 2006: 528–539.

[6] Moran Feldman, Joseph Naor, Roy Schwartz, Justin Ward. Improved Approximations for k-Exchange Systems - (Extended Abstract). *Proc. of ESA*, 2011: 784–798.

[7] Yuli Ye and Allan Borodin. Elimination Graphs. *ACM Transactions on Algorithms*, 2012.