CS 598CSC: Approximation Algorithms	Lecture date: February 11, 2011
Instructor: Chandra Chekuri	Scribe: CC

In the previous lecture we discussed the KNAPSACK problem. In this lecture we discuss other *packing* and *independent set* problems.

1 Maximum Independent Set Problem

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 [1]) *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; simple 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.*

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 $|S| \geq n/(\Delta + 1)$. □

Since the maximum independent set size in a graph is n we obtain the following.

Corollary 3 *Greedy gives a $\frac{1}{\Delta+1}$-approximation for (unweighted)* MIS *in graphs of degree at most Δ.*

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 $\frac{1}{\Delta+1}$-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 4 *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 variables 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 5 *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.

2 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 2 *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 (recall the problem in HW 1) 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 3 *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* $\max_{i,j} A_{ij}/b_i \leq 1/W$.

2.1 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 [2].

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 6 *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 7 $\Pr[\mathcal{E}_1] \leq 1/4$.

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 8 $\Pr[\mathcal{E}_2] \leq 1/2$.

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 9 *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 10 *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 11 *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 12 $\Pr[\mathcal{E}_1(j)] \leq 1/(4k)$.

Claim 13 $\Pr[\mathcal{E}_2(j)] \leq 1/(2k)$.

Lemma 14 *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 15 *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 with 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.

References

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

[2] 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`.