CMPUT 675: Approximation Algorithms **Winter 2005**

Lecture 3: Jan 19

Lecturer: Mohammad R. Salavatipour *Scribe: Jiyang Chen*

3.1 Using Linear Programming (LP) in Design of Approximation Algorithms

In this lecture we will see how linear programming (LP) can be useful in the design of approximation algorithms. Recall that LP is the problem of optimizing (i.e., minimizing or maximizing) a linear function of variables $x_1, \ldots, x_n$ subject to a set of linear inequalities. The standard LP for a minimization problem has the following form:

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^{n} c_i x_i \\ \text{subject to} & \sum_{j=1}^{n} a_{ij} x_j \geq b_i \quad (1 \leq i \leq m) \\ & x_j \geq 0 \end{array}$$

and for a maximization problem:

$$\begin{array}{ll} \text{maximize} & \sum_{i=1}^{n} c_i x_i \\ \text{subject to} & \sum_{j=1}^{n} a_{ij} x_j \leq b_i \quad 1(\leq i \leq m) \\ & x_j \geq 0 \end{array}$$

Definition 3.1 *A feasible solution is a solution (assignment to the variables) that satisfies all the constraints.*

If we require the variable x_i's to be integers (e.g. from $\{0, 1\}$), then we have an Integer Program (IP). If we relax the integrality condition the we obtain an LP which is called the LP-relaxation of the corresponding IP. Solving IP's in general is NP-complete whereas we can solve linear programs in polynomial time (for example by using the Ellipsoid method or one of the interior point methods). We can solve LP's in polynomial time, even if the number of constraints is exponential, as long as there is a polynomial time representation for the LP, and there is a polynomial time separation oracle for the problem:
Separation oracle: Given any assignment to the variables, it tells if it is a feasible solution or not and if not, finds the violated constraint(s).

Consider an IP/LP formulation of a minimization problem. Trivially the optimal solution to the LP is a lower bound for the optimal solution to IP. Similarly, the optimal fractional solution for a maximization problem is an upper bound for the optimal integral solutions. These trivial bounds are sometimes used to bound the approximation ratios of algorithms.

3.1.1 IP/LP Formation for Set Cover

Here is one possible IP formulation of the Set Cover problem. Note that there may be several IP formulations for a problem. We will have one indicator variable x_s for every set S:

$$\begin{array}{lll} \text{minimize} & \sum_{s \in S} c_s x_s & \\ \text{subject to} & \forall e \in U: & \sum_{s: e \in S} x_s \geq 1 \\ & x_s \in \{0,1\} & \end{array}$$

Now relax the integrality constraint to obtain the corresponding LP.

Solution to an LP-relaxation is usually called the *fractional solution* (vs the solution) and is denoted by OPT_f.

Our first algorithm using LP for Set Cover is an f-approximation, where f is the frequency of the most frequent element in U.

3.1.2 An f-approximation using LP Rounding

Consider the IP/LP formulation of Set Cover given above.

- Take the LP relaxation, and solve it.
- Let $\vec{x}^*$ be the optimal fractional solution, pick every set S_j for which $x^*_{S_j} \geq \frac{1}{f}$, (i.e. round $x^*_{S_j}$ up to $\hat{x}_{S_j} = 1$). Otherwise assign $\hat{x}_{S_j} = 0$. So $\hat{x} = (\hat{x}_{S_1}, \ldots, \hat{x}_{S_k})$ is the integral solution.
- return $\hat{x}$.

The following two lemmas show that this algorithm is an f-approximation for the Set Cover problem.

Lemma 3.2 *The solution of this algorithm is a set cover.*

Proof: Let $I = \{j | \hat{x}_{S_j} = 1\}$, i.e. those indices whose corresponding set is picked. By way of contradiction, assume that for some $e \notin U_{j \in I} S_j$. Thus, for each set S_j which contains e: $\hat{x}_{S_j} = 0$. This implies that:

$$x^*_{S_j} < \frac{1}{f} \Rightarrow \sum_{j: e \in S_j} x^*_{S_j} < 1$$

which contradicts the constraint in LP corresponding to e. Therefore, the solution is a set cover. ■

Lemma 3.3 *The approximation ratio of the algorithm is f.*

Proof: For each $\hat{x}_{S_j}$ in the final solution,

$$\hat{x}_{S_j} \leq f \times x^*_{S_j} \Rightarrow \sum_{S \in \mathcal{S}} c_S \hat{x}_S \leq f \cdot \sum c_s x^*_S = f \cdot OPT_f \leq f \cdot OPT.$$

■

3.1.3 Randomized Rounding Algorithm for Set Cover

In this section we give another $O(\log n)$-approximation for Set Cover using a powerful technique called randomized rounding. The general idea is to start with the optimal fractional solution (solution to the LP) and then round the fractional values to 1 with some appropriate probabilities (that often depends on the value of the variable itself).

Here is the algorithm for the Set Cover:

- Take the LP relaxation and solve it.
- For each set S, pick S with probability $P_s = x_s^*$ (i.e. round x_s^* up to 1 with probability x_s^*), let's call the integer value $\hat{x}_s$),

Consider the collection $C = \{S_j \mid \hat{x}_{S_j} = 1\}$:

$$\mathrm{E}[cost(C)] = \sum_{S_i \in \mathcal{S}} \Pr[S_j \text{ is picked}] \cdot c_{S_j} = \sum x^*_{S_j} c_{S_j} = OPT_f \tag{3.1}$$

Let α be large enough, such that $(\frac{1}{e})^{\alpha \log n} \leq \frac{1}{4n}$. Repeat the algorithm above $\alpha \log n$ times and let $C' = \bigcup_{i=1}^{\alpha \log n} C_i$ be the final solution, where C_i is the collection obtained after round i of the algorithm. We will show that with sufficiently large probability C' is a Set Cover and is not larger than the optimal solution by a factor larger than $O(\log n)$.

Suppose that e_j belongs to $S_1, ...S_q$ (for some q and some ordering of the sets). By the constraint for e_j, in any fractional feasible solution:

$$x_{S_1} + x_{S_2} + ...x_{S_q} \geq 1$$

This is true in particular for the optimal fractional solution. It can be shown that the probability that e_j is covered is minimized when

$$\begin{aligned} & x_{S_1} = x_{S_2} = ... = x_{S_q} = \tfrac{1}{q} \\ \Rightarrow\ & \Pr[e_j \text{ is not covered in } C_i] \leq (1 - \tfrac{1}{q})^q < \tfrac{1}{e} \\ \Rightarrow\ & \Pr[e_j \notin C'] \leq (\tfrac{1}{e})^{\alpha \log n} \leq \tfrac{1}{4n} \end{aligned}$$

Sum over all e_j:

$$\Pr[\exists e_j, e_j \notin C',\ \text{(i.e. C' is not a set cover)}] \leq \text{n} \cdot \frac{1}{4\text{n}} \leq \frac{1}{4}$$

Let's call the event "C' is not a Set Cover", $E1$. By above:

$$\Pr[E_1] \leq \frac{1}{4}. \tag{3.2}$$

On the other hand, by (3.1) and by summing over all rounds:

$$\mathrm{E}[cost(C')] \leq \alpha \log n \cdot OPT_f$$

Markov's inequality says for any random variable X: $\Pr[X \geq t] \leq \frac{\mathrm{E}[X]}{t}$. Define the bad event E_2 to be the event that $cost(C') > 4\alpha \log n \cdot OPT$. Thus:

$$\Pr[E_2] = \Pr[cost(C') > 4\alpha \log n \cdot OPT] \leq \frac{\alpha \log n \cdot OPT_f}{4\alpha \log n \cdot OPT_f} \leq \frac{1}{4} \tag{3.3}$$

By (3.2) and (3.3), the probability that either C' is not a set cover (i.e. E_1 happens) or that C' is a set cover with large cost (i.e. E_2 happens) is at most: $\Pr[E_1] + \Pr[E_2] \leq \frac{1}{2}$. Therefore, with probability $\geq \frac{1}{2}$, C' is a set cover with $cost(C') \leq 4\alpha \log n \cdot OPT_f \leq 4\alpha \log n \cdot OPT$. Repeating this t times, the probability of failure at all rounds is at most $\frac{1}{2^t}$. Therefore, the probability of success for at least one run of the algorithm is $1 - \frac{1}{2^t}$. For large enough t, this is arbitrarily close to 1.

3.1.4 Integrality Gap

As we mentioned earlier, comparing the integral solution obtained by the algorithm with the optimal fractional solution gives a bound on approximation factor of the algorithm. But how good/bad this bound can be? Is the optimal integral solution always within some small factor of the optimal fractional solution?

Definition 3.4 *Let I be an instance of a (minimization) problem π and let $OPT_f(I)$ be the cost of optimal fractional solution to I. The integrality gap (sometimes also called the integrality ratio) of this problem is:*

$$max_I \frac{OPT(I)}{OPT_f I}$$

i.e, the supremum of the ratio of the optimal integral and fractional solutions.

Example: Consider th vertex cover problem and let I be K_n (the complete graph on n vertices). The optimal fractional solution is $x = (\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, ..., \frac{1}{2}) \Rightarrow OPT_f = \frac{n}{2}$, while in the optimal integral solution we must pick at least $n-1$ vertices (if there are two vertices not picked then the edge connecting them is not covered). Clearly $n-1$ vertices cover all the edges. Thus, the integrality gap is $2 - \frac{2}{n}$.

Obtaining bounds on the integrality gaps is important. Generally, any rounding algorithm (based on linear programming), which does not consider the structure of the instance cannot have an approximation factor better than the integrality gap. In general, it is usually more difficult to find approximation algorithms with ratio better than the integrality gap. Sometimes this is the border for hardness of the problem.