Lecture 9

Lecturer: Alantha Newman *December 8, 2017*

1 Randomized LP Rounding

Many of the discrete optimization problems that we want to (approximately) solve can be viewed as *hitting set* problems. For example, vertex cover falls into this category. Let $\mathcal{U} = \{e_1, \ldots, e_n\}$ denote a universe of n elements, and let $\mathcal{T} = \{T_1, T_2, \ldots, T_m\}$ denote a family of m sets such that $T_j \subseteq \mathcal{U}$ for all $j \in \{1, 2, \ldots, m\}$. Each element in $\mathcal{U}$ is equipped with a cost function $c : \mathcal{U} \rightarrow \mathbb{R}^+$. The goal is to find a subset $A \subseteq \mathcal{U}$ with minimum cost $c(A)$ such that $T_j \cap A \neq \emptyset$ for all $j \in \{1, 2, \ldots, m\}$. The hitting set problem can be modeled with the following linear program.

$$\begin{aligned} \min \ & \sum_{i=1}^{n} c(e_i) \cdot x_{e_i} \\ \text{subject to:} \ & \sum_{e_i \in T_j} x_{e_i} \geq 1, \quad \text{for all sets } T_j, \\ & x_{e_i} \geq 0. \end{aligned} \tag{$\mathrm{P}_{Hit-Set}$}$$

Suppose that each set T_j has size at most k, i.e. $|T_j| \leq k$. If we consider the solution set $S = \{e_i \in \mathcal{U} \mid x_{e_i} \geq 1/k\}$, then S is a hitting set and the cost of S is at most k times the optimal value of ($\mathrm{P}_{Hit-Set}$). Note that this is a generalization of the 2-approximation algorithm we saw for the vertex cover problem in Lecture 3. However, since k could be very large, this deterministic method might not yield a good approximation guarantee.

The problems that we will study in this lecture and the next can all be viewed as hitting set problems, e.g. maximum satisfiability, set cover and dominating set. A common approach to finding feasible solutions for such problems is to interpret the variables x_{e_i} as probabilities, i.e. the probability of adding element e_i to the solution set. This is the main idea behind *randomized rounding* of linear programs.

2 Maximum Satisfiability

In the maximum satisfiability problem, we are given a formula in conjunctive normal form and the goal is to find a boolean assignment for the variables that satisfies the maximum number of clauses.

Input: A formula F on n variables, $\{x_1, x_2, \ldots, x_n\}$, and m clauses, $\{C_1, \ldots, C_m\}$. Each clause C_j has a nonnegative weight w_j and consists of some subset of positive and negative variables (a.k.a. literals). For example, $F = (x_1 \vee \bar{x_2}) \wedge (x_3) \wedge (x_1 \vee x_2 \vee \bar{x_3}) \wedge \ldots$. (We assume that each clause contains only one copy of each variable.)

Output: An assignment $x_i \in \{0, 1\}$ for each variable maximizing the total weight of the satisfied clauses.

The maximum satisfiability problem can be viewed as a hitting set problem. We can construct a set system whose sets correspond to the clauses in F as well as pairs of literals $\{x_i, \bar{x_i}\}$. A satisfying assignment corresponds to one in which at least one literal from each of these sets is included in the hitting set. We note that the maximum satisfiability problem is NP-hard since it generalizes the decision problem 3-SAT: If m is the maximum number of clauses that can be satisifed, then the formula is satisfiable. If each clause contains exactly two literals, then this problem is known as MAX-2-SAT and is NP-hard

(even though the decision problem has an efficient algorithm). We now consider a simple randomized algorithm for the maximum satisfiability problem.

ALGORITHM 1

For $i = 1$ to n:

Set each variable $x_i \to 1$ with probability $\frac{1}{2}$.

The expected value of the assignment output by the algorithm is:

$$\mathbb{E}[W] \quad = \quad \sum_{j=1}^{m} w_j \cdot \Pr[C_j \text{ is satisfied}].$$

Lemma 1. *If the clause C_i has size k (i.e. k variables), then:*

$$\Pr[C_j \textit{ is satisfied}] = 1 - \frac{1}{2^k}.$$

Proof. $\Pr[C_j \text{ is not satisfied}] = \frac{1}{2^k}$. So $\Pr[C_j \text{ is satisfied}] = 1 - \frac{1}{2^k}$. □

If all clauses have size ≥ 2, ALGORITHM 1 has an approximation guarantee of $\frac{3}{4}$. But can we achieve an approximation ratio of $\frac{3}{4}$ when there are also unit clauses? ALGORITHM 1 has good performance when the clause lengths are long. We now consider another algorithm that performs well when the clauses are short and show that combining these two algorithms yields a $\frac{3}{4}$-approximation algorithm.

2.1 LP Relaxation for Maximum Satisfiability

For each clause C_j, we let S_j^+ denote the positive variables and S_j^- denote the negated variables. The following integer program exactly models (i.e. is equivalent to) the maximum satisfiability problem.

$$\begin{aligned} & \max \sum_{j=1}^{m} w_j z_j \\ \text{subject to: } & \sum_{i \in S_j^+} y_i + \sum_{i \in S_j^-} (1 - y_i) \geq z_j, \quad \text{for } j \in \{1, 2, \ldots, m\}, \\ & y_i, z_j \in \{0, 1\}. \end{aligned}$$

We consider the corresponding linear programming relaxation. Now the variable z_j denotes the fraction of clause C_j that is satisfied and y_i is the fractional truth value of variable x_i.

$$\begin{aligned} & \max \sum_{j=1}^{m} w_j z_j \\ \text{subject to: } & \sum_{i \in S_j^+} y_i + \sum_{i \in S_j^-} (1 - y_i) \geq z_j, \quad \text{for } j \in \{1, 2, \ldots, m\}, \\ & 0 \leq y_i \leq 1, \\ & 0 \leq z_j \leq 1. \end{aligned} \tag{$P_{Max-Sat}$}$$

ALGORITHM 2

Find an optimal solution $\{\mathbf{y}^*, \mathbf{z}^*\}$ for $(P_{Max-Sat})$.
For $i = 1$ to n:

Set each variable $x_i \to 1$ with probability y_i^*.

To analyze the performance of ALGORITHM 2, we need some useful inequalities.

Fact 2. [Arithmetic-geometric mean inequality] *For any nonnegative $a_1, \ldots, a_k$,*

$$\left(\prod_{i=1}^{k} a_i\right)^{\frac{1}{k}} \quad \leq \quad \frac{1}{k}\sum_{i=1}^{k} a_i.$$

Fact 3. *If a function $f(x)$ is concave on the interval $[0,1]$ (that is, $f''(x) \leq 0$ on $[0,1]$), and $f(0) = 0$ and $f(1) = b$, then $f(x) \geq bx$ for $x \in [0,1]$.*

Theorem 4. ALGORITHM 2 *is a $(1 - \frac{1}{e})$-approximation for maximum satisfiability.*

Proof. What is the probability that clause C_j is satisfied by the assignment given by ALGORITHM 2? Let ℓ_j denote the length of clause C_j.

$$\begin{aligned}
\Pr[C_j \text{ is not satisfied}] &= \prod_{i \in S_j^+} (1 - y_i^*) \prod_{i \in S_j^-} y_i^* \\
&\leq \left[\frac{1}{\ell_j}\left(\sum_{i \in S_j^+} (1 - y_i^*) + \sum_{i \in S_j^-} y_i^*\right)\right]^{\ell_j} \qquad (1) \\
&= \left[1 + \frac{1}{\ell_j}\left(-\ell_j + \sum_{i \in S_j^+} (1 - y_i^*) + \sum_{i \in S_j^-} y_i^*\right)\right]^{\ell_j} \\
&= \left[1 - \frac{1}{\ell_j}\left(\sum_{i \in S_j^+} y_i^* + \sum_{i \in S_j^-} (1 - y_i^*)\right)\right]^{\ell_j} \\
&\leq \left[1 - \frac{1}{\ell_j} z_j^*\right]^{\ell_j}.
\end{aligned}$$

Line (1) follows from Fact 2. Thus, we have:

$$\Pr[C_j \text{ is not satisfied}] \quad \leq \quad \left[1 - \frac{1}{\ell_j} z_j^*\right]^{\ell_j}.$$

This implies:

$$\begin{aligned}
\Pr[C_j \text{ is satisfied}] &\geq 1 - \left[1 - \frac{1}{\ell_j} z_j^*\right]^{\ell_j} \\
&\geq \left[1 - \left(1 - \frac{1}{\ell_j}\right)^{\ell_j}\right] z_j^*. \qquad (2)
\end{aligned}$$

Line (2) follows from Fact 3. So we can compute the expected value of the solution:

$$\begin{aligned}
\mathbb{E}[W] &= \sum_{j=1}^{m} w_j \cdot \Pr[C_j \text{ is satisfied}] \\
&\geq \sum_{j=1}^{m} w_j \cdot z_j^* \left[1 - \left(1 - \frac{1}{\ell_j}\right)^{\ell_j}\right] \\
&\geq \min_{k \geq 1} \left[1 - \left(1 - \frac{1}{k}\right)^{k}\right] \sum_{j=1}^{m} w_j \cdot z_j^* \\
&\geq \min_{k \geq 1} \left[1 - \left(1 - \frac{1}{k}\right)^{k}\right] \cdot OPT.
\end{aligned}$$

Using the fact that $1 + x \leq e^x$ for all x, we have:

$$\begin{aligned}
1 - \frac{1}{k} &\leq \frac{1}{e^{\frac{1}{k}}} \quad \Rightarrow \\
\left(1 - \frac{1}{k}\right)^{k} &\leq \frac{1}{e}.
\end{aligned}$$

This implies:

$$\min_{k \geq 1} \left[1 - \left(1 - \frac{1}{k}\right)^{k}\right] \geq \left(1 - \frac{1}{e}\right),$$

which implies that $\mathbb{E}[W] \geq (1 - \frac{1}{e}) \cdot OPT$. □

Now we combine both algorithms. Namely, we run both algorithms and output the assignment that satisfies the most clauses. To analyze this, we consider the expected value of an assignment if we run each algorithm with probability $\frac{1}{2}$. (Note that we can run Algorithm 1 with probability $\alpha \geq 0$ and Algorithm 2 with probability $\beta \geq 0$ as long as $\alpha + \beta = 1$.) Let W_1 and W_2 denote the outputs of Algorithm 1 and Algorithm 2, respectively.

$$\begin{aligned}
\mathbb{E}\left[\max\{W_1, W_2\}\right] &\geq \mathbb{E}\left[\frac{W_1}{2} + \frac{W_2}{2}\right] = \mathbb{E}\left[\frac{W_1}{2}\right] + \mathbb{E}\left[\frac{W_2}{2}\right] \\
&\geq \sum_{j=1}^{m} w_j \cdot z_j^* \left[\frac{1}{2}\left(1 - \left(1 - \frac{1}{\ell_j}\right)^{\ell_j}\right) + \frac{1}{2}\left(1 - \frac{1}{2^{\ell_j}}\right)\right].
\end{aligned}$$

Proving the following claim proves that the best of the two algorithms is a $\frac{3}{4}$-approximation algorithm.

Claim 5.

$$\left[\frac{1}{2}\left(1 - \left(1 - \frac{1}{\ell_j}\right)^{\ell_j}\right) + \frac{1}{2}\left(1 - \frac{1}{2^{\ell_j}}\right)\right] \geq \frac{3}{4}.$$

Proof. Note that ℓ_j is always an integer with value at least one. For $\ell_j = 1$, we can see that the claim is true. Similarly for $\ell_j = 2$. For $\ell_j \geq 3$, the value is actually at least .753. □

Remark: There is a randomized algorithm achieving an approximation ratio of $\frac{3}{4}$ for the maximum satisfiability problem that does *not* use a linear program [PS11]. There is also a deterministic rounding algorithm that achieves an approximation guarantee of $\frac{3}{4}$ and *does* use a linear program [VZ11]. However, it is not known whether or not there is a $\frac{3}{4}$-approximation algorithm that is both combinatorial and deterministic.

References

[GW94] Michel X. Goemans and David P. Williamson. New 3/4-approximation algorithms for the maximum satisfiability problem. *SIAM Journal on Discrete Mathematics*, 7(4):656–666, 1994.

[PS11] Matthias Poloczek and Georg Schnitger. Randomized variants of Johnson's algorithm for MAX SAT. In *Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms*, pages 656–663. SIAM, 2011.

[Vaz13] Vijay V. Vazirani. *Approximation Algorithms*. Springer, 2013.

[VZ11] Anke Van Zuylen. Simpler 3/4-approximation algorithms for MAX SAT. In *International Workshop on Approximation and Online Algorithms*, pages 188–197. Springer, 2011.

[WS11] David P. Williamson and David B. Shmoys. *The Design of Approximation Algorithms*. Cambridge University Press, 2011.

These lecture notes are based on Chapter 5 of [WS11] and Chapter 16 of [Vaz13]. The original presentation of the algorithm for maximimum satisfiability can be found in [GW94].