MS&E-212: Combinatorial Optimization
Instructor: Professor Amin Saberi (saberi@stanford.edu)
Thursday 2/21/19

Lecture 12: Approximation Algorithms II

In this lecture, we continue our discussion on approximation algorithms. In particular, we focus on techniques to define an appropriate *relaxed problem* that can be solved in polynomial time and then to convert the solution to an appropriate integral one.

1 Minimum Weighted Vertex Cover

Given a graph $G(V, E)$, and a weight function $W : V \mapsto \mathbb{R}^+$, find a subset $S \subseteq V$ that covers all the edges and has the minimum $W(S)$, where $W(S) = \sum_{v \in S} w(v)$.

We can formulate this problem as an integer program as follows: associate variable x_v with node v,

$$\begin{aligned} \textbf{minimize:} \quad & \sum_{v \in V} w(v) x_v \\ \textbf{Constraint 1:} \quad & x_u + x_v \geq 1, \ (u, v) \in E \\ \textbf{Constraint 2:} \quad & x_v \in \{0, 1\} \end{aligned}$$

Since the problem is **NP-C**, we first attempt to solve a simpler problem for which polynomial-time algorithms exists: we modify Constraint 2 to be $0 \leq x_v \leq 1$. Let $\{x_v^*, \ v \in V\}$ be the solution of the LP. We need to convert this fractional solution to an integral one; we use the following rounding policy: if $x_v^* < 0.5$ then we set $\widehat{x}_v = 0$ otherwise we set $\widehat{x}_v = 1$.

Fact: $\{\widehat{x}_v, \ v \in V\}$ is a feasible solution. Because $\{x_v^*, \ v \in V\}$ is a feasible solution of the LP, $x_v^* + x_u^* \geq 1$ thus $x_v^* \geq 1/2$ or $x_u^* \geq 1/2$ which implies that $\widehat{x}_v + \widehat{x}_u \geq 1$.

Lemma 1 *Let $S = \{v \in V : \ \widehat{x}_v = 1\}$, we have:*

$$\sum_{v \in S} w(v) \leq 2S^*$$

where S^* is the optimum solution.

Proof: Since the feasible region of the IP problem is a subset of the feasible set of the LP one, the optimum solution of the LP is a lower bound for the optimum solution of the IP. Moreover, note that $\widehat{x}_v \leq 2x_v^*$, thus

$$\sum_{v \in V} w(v) x_v^* \leq \sum_{v \in S^*} \widehat{x}_v \leq \sum_{v \in S} \widehat{x}_v \leq 2 \sum_{v \in V} w(v) x_v^*$$

■

Therefore, rounding the LP solution gives a factor 2 approximation algorithm.

2 Maximum Cut

Recall the definition of a graph cut we saw earlier in the quarter. Given graph $G(V, E)$, we define edge weights $w_{ij} = w_{ji}$ such that $w_{ij} > 0$ whenever $(i, j) \in E$ and $w_{ij} = 0$ otherwise. Given a set $S \subseteq V$, the cut value obtained by partitioning the vertices into S and its complement $\bar{S}$ is given by

$$\text{cut}(S, \bar{S}) = \sum_{i \in S,\, j \in \bar{S}} w_{ij} \ .$$

We saw a (relatively) simple randomized algorithm may be used to find the value of the minimum cut. Now suppose we wish to find the value of the maximum cut of G. This problem is in fact NP-hard. The max-cut problem is to find S that maximizes $\text{cut}(S, \bar{S})$.

We will provide a very simple randomized algorithm that gives a $\frac{1}{2}$-approximation in expectation.

Algorithm 1 A randomized algorithm for max-cut

$S \leftarrow \emptyset$.
for $i = 1$ to n **do**
  Independently flip a coin for v_i. If heads, place in S, otherwise don't.
end for
Output $\text{cut}(S, \bar{S})$.

Then each node will be in the set S wp 1/2. Now we can look at the probability that each edge is cut: an edge is cut if its enpoints are separated by the cut, which happens with probability exactly 1/2. The expected output of algorithm 1 then is. where X_{ij} is the indicator variable that edge (i, j) is cut:

$$\begin{aligned} E[mboxcut(S, \bar{S})] &= E\left[\sum_{(i,j) \in E} w_{ij} \cdot X_{ij}\right] \\ &= \sum_{(i,j) \in E} w_{ij} \cdot E[X_{ij}] \\ &= \frac{1}{2} \cdot \sum_{(i,j) \in E} w_{ij}. \end{aligned}$$

Finally, note that any max-cut must be bounded by the total sum of the weights, i.e. $\text{cut}(S, \bar{S}) \leq \sum_{(i,j)} w_{ij}$ for all S, so we immediately get that our algorithm gives a 1/2-approximation in expectation.

3 Maximum Satisfiability

We return to the setting of boolean formulas and consider a problem related to satisfiability: for a given formula in Conjunctive Normal Form (CNF), what is the maximum number of its clauses that can be satisfied by assigning true and false value to variables? More concretely, suppose we have n variables $x_1, ..., x_n$ and m clauses $C_1, ..., C_m$ where

$$C_i = \left(\bigvee_{i \in S_i^+} x_i\right) \vee \left(\bigvee_{i \in S_i^-} \bar{x}_i\right).$$

The **maximum satisfiability problem** is to find the maximum number of clauses that may be satisfied by an assignment x.

We first propose a simple randomized algorithm to approximate a solution to this problem. Set each x_i independently to be 0 or 1 with probability 1/2. The probability that C_i is satisfied by this assignment is

$1 - 2^{|C_i|}$ for every i. If we let Z_i denote the event that clause C_i is satisfied by this random assignment and $Z = \sum_{i=1}^{m} Z_i$ be the total number of satisfied clauses, we may compute:

$$\mathrm{E}[Z] = \sum_{i=1}^{m} \mathrm{E}[Z_i] = \sum_{i=1}^{m} \left(1 - 2^{|C_i|}\right).$$

In the case that all of our clauses are large, i.e. $|C_i| \geq K$ for each i, then this randomized algorithm has an approximation ratio of $\geq 1 - 2^{-K}$ in expectation:

$$m(1 - 2^{-K}) \leq \mathrm{E}[Z] \leq OPT \leq m.$$

Having a bound on the approximation ratio of the algorithm in expectation is often unsatisfactory. This is because a bound on expected value does not bound the probability that the algorithm returns a good solution. Concentration inequalities can help us estimate such probabilities, but in some cases we may do even better. This algorithm may be **derandomized** using conditional expectation as follows.

Algorithm 2 Derandomized Approximation Algorithm for Maximum Satisfiability

for $i = 1$ to n **do**
  Compute $\frac{1}{2}\mathrm{E}[Z \mid x_i = 1, x_{i-1}, ..., x_1]$ and $\frac{1}{2}\mathrm{E}[Z \mid x_i = 0, x_{i-1}, ..., x_1]$.
  Set $x_i = 1$ if the first expression is larger than the second, set $x_i = 0$ otherwise.
end for
return x

For motivation, we consider the first step of the algorithm. Note that

$$\mathrm{E}[Z] = \frac{1}{2}\mathrm{E}[Z \mid x_1 = 1] + \frac{1}{2}\mathrm{E}[Z \mid x_1 = 0].$$

Both terms on the right hand side may be computed simply by summing over all clauses the probability that C_i is satisfied given the information on x_1. The equation above implies that $\mathrm{E}[Z \mid x_1 = 1] \geq \mathrm{E}[Z]$ or $\mathrm{E}[Z \mid x_1 = 0] \geq \mathrm{E}[Z]$. Thus if we choose the greater expectation in each step of the algorithm, we will deterministically build up an assignment x such that

$$\mathrm{E}[Z|x] \geq \mathrm{E}[Z] \geq m(1 - 2^{-K})$$

where $\mathrm{E}[Z|x]$ is the number of clauses x satisfies.

The approximation ratio of algorithm 2 is good only if the clauses have a large number of variables. We present a different algorithm for dealing with the possibility that some of the clauses may be small. It is based on the now familiar concept of LP relaxation. We write down an integer program for maximum satisfiability.

$$\begin{aligned}
\textbf{maximize:} \quad & \sum_{i=1}^{m} q_i & \\
\textbf{s.t.} \quad & q_i \leq \sum_{j \in S_i^+} y_j + \sum_{j \in S_i^-} (1 - y_j) & \forall\, i \\
& q_i, y_j \in \{0, 1\} & \forall\, i, j
\end{aligned}$$

The variables q_i correspond to the truth value of each clause C_i, and the variables y_j correspond to the values of each boolean variable x_i. We relax the last condition to be $0 \leq q_i, y_j \leq 1$ in order to get a linear program.

Algorithm 3 An LP Rounding Algorithm for Maximum Satisfiability

Solve the LP given above.
for $j = 1$ to n **do**

$$\text{Independently set } x_j = \begin{cases} 1 & : \text{with probability } y_j^* \\ 0 & : \text{with probability } 1 - y_j^* \end{cases}$$

end for

In order to analyze algorithm 3 we consider the probability that a particular clause is satisfied; Let u focus on one clause, say C_1 and assume without loss of generality that $C_1 = x_1 \vee ... \vee x_k$. We have $q_1^* = \min\{y_1^* + ... + y_k^*, 1\}$ and by the inequality of arithmetic and geometric means:

$$\begin{aligned} Pr[C_1] &= 1 - \prod_{j=1}^{k}(1 - y_j^*) \\ &\geq 1 - \left(\frac{1}{k}\sum_{j=1}^{k}(1 - y_j^*)\right)^k \\ &\geq 1 - \left(1 - \frac{q_1^*}{k}\right)^k \\ &\geq q_1\left(1 - \left(1 - \frac{1}{k}\right)^k\right) \\ &\geq q_1(1 - 1/e) \end{aligned}$$

This last line implies, through the linearity of expectation, that this rounding procedure gives a $(1 - 1/e)$-approximation for maximum satisfiability regardless of the size of the smallest clause. Algorithm 3 may also be derandomized by the method of conditional expectations.

These two derandomized algorithms may be combined to give a factor 3/4 approximation algorithm for maximum satisfiability. We simply run both algorithms on a given problem instance and output the solution with the better value. This procedure itself may be viewed as a derandomization of an algorithm that flips a fair coin to decide which randomized sub-algorithm to run. That algorithm has approximation factor 3/4:

$$\mathrm{E}[Z] = \sum_{i=1}^{m}\mathrm{E}[Z_i] \geq \sum_{i=1}^{m}\frac{1}{2}\left((1 - 2^{-|C_i|}) + \left(1 - \left(1 - \frac{1}{|C_i|}\right)^{|C_i|}\right)\right) \geq (3/4)m.$$