Advanced Algorithms (VII)

Shanghai Jiao Tong University

Chihao Zhang

April 20, 2020

The Probabilistic Method

In the class of Combinatorics, you already learnt the *probabilistic method*

CS477 Combinatorics (Spring 2019)

11. Lecture 11, 2019/05/17. Ramsey problem on integers. Erdos' proof of Ramsey lower bound.
12. Lecture 12, 2019/05/26. The probabilistic method.
13. Lecture 13, 2019/06/14. Erdos' theorem on large chromatic numbers and large girth. Set families. Some combin

This is an important technique to prove the existence of some object.

Sometimes, it is also useful to “find the object”

Max Cut

Given an undirected graph $G = (V, E)$, the max cut of G is the partition $V = S \cup \bar{S}$ such that $|E(S, \bar{S})|$ is maximized

edge between S and $\bar{S}$

It is NP-hard to determine the max cut exactly

On the other hand, each graph contains a cut of size at least $\dfrac{|E|}{2}$

We find a partition $(S, \bar{S})$ by tossing a fair coin at each vertex v

If the coin gives HEAD, we put v in S, otherwise, put v in $\bar{S}$

We can compute

$$\mathbf{E}[\,|E(S,\bar{S})|\,] = \sum_{e \in E} \Pr[e \text{ is in the cut}] = \frac{|E|}{2}.$$

So there exists a cut of size at least $\dfrac{|E|}{2}$

Can we turn the existence proof into an algorithm?

The following straightforward strategy turns the argument into a *Las-Vegas algorithms*

"Repeat tossing coins until $|E(S, \bar{S})| \geq \dfrac{|E|}{2}$*"*

We know $\mathbf{E}[\,|E(S, \bar{S})|\,] = \dfrac{|E|}{2}$, so what is the expected running time of the algorithm?

Let p be the probability that our algorithm terminates in one round

Namely $p = \Pr\left[|E(S,\bar{S})| \geq \frac{m}{2}\right]$ where $m = |E|$.

Then

$$\frac{m}{2} = \mathbf{E}[|E(S,\bar{S})|] = \sum_{i=0}^{m} i \cdot \Pr[|E(S,\bar{S})| = i]$$

$$\leq \left(\frac{m}{2} - 1\right)(1-p) + pm$$

So $p \geq \dfrac{2}{m+2}$

So we obtained a polynomial-time randomized approximation algorithm with approximation ratio $\frac{1}{2}$

Approximation Ratio of an algorithm A

for maximization problem:

$$\alpha(A) = \min_G \frac{A(G)}{\mathrm{OPT}(G)}$$

for minimization problem:

$$\alpha(A) = \max_G \frac{A(G)}{\mathrm{OPT}(G)}$$

Derandomization

Our algorithm can be de-randomized using the method of conditional expectation

Fix an order of vertices $\{v_1, v_2, \ldots, v_n\}$

Let the coins be $X_1, X_2, \ldots, X_n$

We will decompose $\mathbf{E}[\,|E(S, \bar{S})|\,]$ using conditional expectation

$$
\begin{aligned}
\mathbf{E}[\,|E(S,\bar{S})|\,] &= \mathbf{E}[\mathbf{E}[\,|E(S,\bar{S})|\,|X_1, X_2, \ldots, X_n]] \\
&= \frac{1}{2}\cdot \mathbf{E}[\mathbf{E}[\,|E(S,\bar{S})|\,|\,{\color{red}X_1 = 0}, X_2, \ldots, X_n]] \\
&\quad + \frac{1}{2}\cdot \mathbf{E}[\mathbf{E}[\,|E(S,\bar{S})|\,|\,{\color{red}X_1 = 1}, X_2, \ldots, X_n]] \\
&= \frac{1}{2}\cdot \underbrace{\boxed{\mathbf{E}[\,|E(S,\bar{S})|\,|\,X_1 = 0]]}}_{E_0} + \frac{1}{2}\cdot \underbrace{\boxed{\mathbf{E}[\,|E(S,\bar{S})|\,|\,X_1 = 1]]}}_{E_1}
\end{aligned}
$$

So we know at least one of $E_0 \geq \dfrac{m}{2}$ and $E_1 \geq \dfrac{m}{2}$ holds

Moreover, both E_0 and E_1 can be efficiently computed

We can set $X_1 = 0$ or $X_1 = 1$ according to which of E_0 and E_1 is bigger

The argument can proceed until $(S, \bar{S})$ is revealed, deterministically!

In fact, the “derandomized algorithm” is equivalent to a simple greedy strategy

We obtained the approximation ratio of the greedy algorithm as a byproduct

Max SAT

The simple "Tossing Coins" strategy can also be applied to the MAXimum SATisfiability problem.

MaxSAT

Input: A CNF formula $\phi = C_1 \wedge C_2 \cdots \wedge C_m$

Problem: Compute an assignment that satisfies maximum number of clauses

Formula ϕ, variables $V = \{x_1, \ldots, x_n\}$, $|C_i| = \ell_i \geq 1$

Let us analyze the “tossing fair coins” algorithm

Let X be the number of satisfied clauses

$$\mathbf{E}[X] = \sum_{i=1}^{m} \Pr[C_i \text{ is satisfied}] = \sum_{i=1}^{m} \left(1 - 2^{-\ell_i}\right) \geq \frac{m}{2}$$

Recall

Approximation Ratio of an algorithm A

for maximization problem:

$$\alpha(A) = \min_{G} \frac{A(G)}{\mathrm{OPT}(G)}$$

To bound the approximation ratio, we need an upper bound for $\mathrm{OPT}(\phi)$

A trivial upper bound is $\mathrm{OPT}(\phi) \leq m$

So the approximation ratio is 0.5

Can we improve it?

In the analysis $\displaystyle\sum_{i=1}^{m}\left(1-2^{-\ell_i}\right) \geq \frac{m}{2}$ we use $\ell_i \geq 1$

In fact, we can tweak those singleton clauses

If for some $x \in V$, only one of x and $\bar{x}$ is in ϕ

- we can toss an unfair coin to increase its chance to be satisfied

If both x and $\bar{x}$ are in ϕ,

- only one of them can be satisfied in any assignment!

Both cases are good for us!

Assume there are more positive singletons than negative singletons in ϕ

Let $S = \{x \in V : \text{both } x \text{ and } \bar{x} \text{ are clauses}\}$ and $t = |S|$

Then $\mathrm{OPT}(\phi) \leq m - t$

Let $\mathscr{C}$ be the set of clauses and

$$\mathscr{C}' = \mathscr{C} \backslash \{\text{singleton } x \text{ and } \bar{x} \text{ with } x \in S\}$$

For all $\bar{x} \in \mathscr{C}'$, change it to x

Switch the positive and the negative for all appearance of x

$$\mathbf{E}[X] = t + \sum_{C \in \mathscr{C}'} \Pr[C \text{ is satisfied}] \geq t + (m - 2t) \min\{p, 1 - p^2\}$$

The term $\min\{p, 1 - p^2\}$ is because the worst case now is either a positive singleton x or $\bar{y} \vee \bar{z}$

Therefore

$$\mathbf{E}[X] \geq t + (\mathrm{OPT} - t) \min\{p, 1 - p^2\} \geq \min\{p, 1 - p^2\} \cdot \mathrm{OPT}$$

For $p = 1 - p^2$, we have a 0.618-approximation algorithm

Non-identical Coins via LP

The drawback of previous algorithms is that we toss the same coin for each variable

The *linear programming* can helps us to choose coins!

We first treat MaxSAT problem as an *integer programming*

The Integer Program

$$
\begin{aligned}
\max \quad & \sum_{j=1}^{m} z_j \\
\text{subject to} \quad & \sum_{i \in P_j} y_i + \sum_{k \in N_j} (1 - y_k) \geq z_j, \quad \forall j \in [m] \text{ s.t. } C_j = \bigvee_{i \in P_j} x_i \vee \bigvee_{k \in N_j} \bar{x}_k \\
& z_j \in \{0,1\}, \quad \forall j \in [m] \\
& y_i \in \{0,1\}, \quad \forall i \in [n]
\end{aligned}
$$

z_j - for each clause C_j

y_i - for each variable x_i

It is NP-hard to solve the IP

The Linear Program

$$\begin{aligned}
\max \quad & \sum_{j=1}^{m} z_j \\
\text{subject to} \quad & \sum_{i \in P_j} y_i + \sum_{k \in N_j} (1 - y_k) \geq z_j, \quad \forall j \in [m] \text{ s.t. } C_j = \bigvee_{i \in P_j} x_i \vee \bigvee_{k \in N_j} \bar{x}_k \\
& 0 \leq z_j \leq 1, \quad \forall j \in [m] \\
& 0 \leq y_i \leq 1, \quad \forall i \in [n]
\end{aligned}$$

$\mathbf{z}^* = \{z_j^*\}_{j \in [m]}, \mathbf{y}^* = \{y_i^*\}_{i \in [n]}$ - the optimal solution of the LP

We toss y_i^*-coin for the variable x_i!

$$\mathrm{OPT}(\phi) \leq \mathrm{OPT}(LP) = \sum_{j=1}^{m} z_j^*$$

$$
\begin{aligned}
\Pr[C_j \text{ is not satisfied}] &= \prod_{i \in P_j} (1 - y_i^*) \prod_{k \in N_j} y_k^* \\
\boxed{\text{AM-GM}} \qquad &\le \left(\frac{1}{\ell_j} \left(\sum_{i \in P_j} (1 - y_i^*) + \sum_{k \in N_j} y_k^* \right) \right)^{\ell_j} \\
&= \left(\frac{1}{\ell_j} \left(\ell_j - \left(\sum_{i \in P_j} y_i^* + \sum_{k \in N_j} (1 - y_k^*) \right) \right) \right)^{\ell_j} \\
&\le \left(1 - \frac{z_j^*}{\ell_j} \right)^{\ell_j}.
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{E}[X] &= \sum_{j=1}^{m} \Pr[C_j \text{ is satisfied}] \\
&\geq \sum_{j=1}^{m} \left(1 - \left(1 - \frac{z_j^*}{\ell_j} \right)^{\ell_j} \right) \\
&\geq \sum_{j=1}^{m} \left(1 - \left(1 - \frac{1}{\ell_j} \right)^{\ell_j} \right) z_j^* \\
&\geq (1 - e^{-1}) \sum_{j=1}^{m} z_j^* \geq \left(1 - \frac{1}{e} \right) \mathrm{OPT}
\end{aligned}
$$

Concavity

≈ 0.632