Chapter 47

The Probabilistic Method

By Sariel Har-Peled, April 5, 2022

“Shortly after the celebration of the four thousandth anniversary of the opening of space, Angary J. Gustible discovered Gustible’s planet. The discovery turned out to be a tragic mistake. Gustible’s planet was inhabited by highly intelligent life forms. They had moderate telepathic powers. They immediately mind-read Angary J. Gustible’s entire mind and life history, and embarrassed him very deeply by making up an opera concerning his recent divorce.”

Gustible’s Planet, Cordwainer Smith

47.1. Introduction

The probabilistic method is a combinatorial technique to use probabilistic algorithms to create objects having desirable properties, and furthermore, prove that such objects exist. The basic technique is based on two basic observations:

1. If $\mathbb{E}[X] = \mu$, then there exists a value $x$ of $X$, such that $x \geq \mathbb{E}[X]$.

2. If the probability of event $\mathcal{E}$ is larger than zero, then $\mathcal{E}$ exists and it is not empty.

The surprising thing is that despite the elementary nature of those two observations, they lead to a powerful technique that leads to numerous nice and strong results. Including some elementary proofs of theorems that previously had very complicated and involved proofs.

The main proponent of the probabilistic method, was Paul Erdős. An excellent text on the topic is the book by Noga Alon and Joel Spencer [AS00].

This topic is worthy of its own course. The interested student is refereed to the course “Math 475 — The Probabilistic Method”.

47.1.1. Examples

47.1.1.1. Max cut

Computing the maximum cut (i.e., max cut) in a graph is a NP-Complete problem, which is APX-Hard (i.e., no better than a constant approximation is possible if $P \neq NP$). We present later on a better approximation algorithm, but the following simple algorithm already gives a pretty good approximation.

Theorem 47.1.1. For any undirected graph $G = (V, E)$ with $n$ vertices and $m$ edges, there is a partition of the vertex set $V$ into two sets $S$ and $T$, such that $|(S, T)| = |\{uv \in E \mid u \in S \text{ and } v \in T\}| \geq \frac{m}{2}$. One can compute a partition, in $O(n)$ time, such that $\mathbb{E}[|(S, T)|] = m/2$.\footnote{This work is licensed under the Creative Commons Attribution-Noncommercial 3.0 License. To view a copy of this license, visit http://creativecommons.org/licenses/by-nc/3.0/ or send a letter to Creative Commons, 171 Second Street, Suite 300, San Francisco, California, 94105, USA.}
**Proof:** Consider the following experiment: randomly assign each vertex to $S$ or $T$, independently and equal probability.

For an edge $e = uv$, the probability that one endpoint is in $S$, and the other in $T$ is $1/2$, and let $X_e$ be the indicator variable with value 1 if this happens. Clearly,

$$
\mathbb{E}\left[\left|\{(u, v) \in S \times T \cup T \times S\}\right|\right] = \sum_{e \in E(G)} \mathbb{E}[X_e] = \sum_{e \in E(G)} \frac{1}{2} = \frac{m}{2}.
$$

Thus, there must be an execution of the algorithm that computes a cut that is at least as large as the expectation -- namely, a partition of $V$ that satisfies the realizes a cut with $\geq m/2$ edges. 

### 47.2. Maximum Satisfiability

In the **MAX-SAT** problem, we are given a binary formula $F$ in **CNF** (Conjunctive normal form), and we would like to find an assignment that satisfies as many clauses as possible of $F$, for example $F = (x \lor y) \land (\overline{x} \lor z)$. Of course, an assignment satisfying all the clauses of the formula, and thus $F$ itself, would be even better -- but this problem is of course NPC. As such, we are looking for how well can be we do when we relax the problem to maximizing the number of clauses to be satisfied.

**Theorem 47.2.1.** For any set of $m$ clauses, there is a truth assignment of variables that satisfies at least $m/2$ clauses.

**Proof:** Assign every variable a random value. Clearly, a clause with $k$ variables, has probability $1 - 2^{-k}$ to be satisfied. Using linearity of expectation, and the fact that every clause has at least one variable, it follows, that $\mathbb{E}[X] = m/2$, where $X$ is the random variable counting the number of clauses being satisfied. In particular, there exists an assignment for which $X \geq m/2$. 

For an instant $I$, let $m_{opt}(I)$, denote the maximum number of clauses that can be satisfied by the “best” assignment. For an algorithm $\text{Alg}$, let $m_{\text{Alg}}(I)$ denote the number of clauses satisfied computed by the algorithm $\text{Alg}$. The **approximation factor** of $\text{Alg}$, is $m_{\text{Alg}}(I)/m_{\text{opt}}(I)$. Clearly, the algorithm of Theorem 47.2.1 provides us with 1/2-approximation algorithm.

For every clause, $C_j$ in the given instance, let $z_j \in \{0, 1\}$ be a variable indicating whether $C_j$ is satisfied or not. Similarly, let $x_i = 1$ if the $i$th variable is being assigned the value TRUE. Let $C_j^+$ be indices of the variables that appear in $C_j$ in the positive, and $C_j^-$ the indices of the variables that appear in the negative. Clearly, to solve **MAX-SAT**, we need to solve:

\[
\begin{align*}
\text{max} & \quad \sum_{j=1}^{m} z_j \\
\text{subject to} & \quad \sum_{i \in C_j^+} x_i + \sum_{i \in C_j^-} (1 - x_i) \geq z_j \quad \text{for all } j \\
& \quad x_i, z_j \in \{0, 1\} \quad \text{for all } i, j
\end{align*}
\]

We relax this into the following linear program:
\[
\begin{align*}
\text{max} & \quad \sum_{j=1}^{m} z_j \\
\text{subject to} & \quad 0 \leq y_i, z_j \leq 1 \text{ for all } i, j \\
& \quad \sum_{i \in C_j} y_i + \sum_{i \in C_j} (1 - y_i) \geq z_j \text{ for all } j.
\end{align*}
\]

Which can be solved in polynomial time. Let \( \hat{t} \) denote the values assigned to the variable \( t \) by the linear-programming solution. Clearly, \( \sum_{j=1}^{\tilde{m}} \tilde{z}_j \) is an upper bound on the number of clauses of \( I \) that can be satisfied.

We set the variable \( y_i \) to 1 with probability \( \hat{y}_i \). This is an instance randomized rounding.

**Lemma 47.2.2.** Let \( C_j \) be a clause with \( k \) literals. The probability that it is satisfied by randomized rounding is at least \( \beta_k \tilde{z}_j \geq (1 - 1/e) \tilde{z}_j \), where

\[
\beta_k = 1 - \left(1 - \frac{1}{k}\right)^k \approx 1 - \frac{1}{e}.
\]

**Proof:** Assume \( C_j = y_1 \lor y_2 \lor \ldots \lor y_k \). By the LP, we have \( \hat{y}_1 + \ldots + \hat{y}_k \geq \tilde{z}_j \). Furthermore, the probability that \( C_j \) is not satisfied is \( \prod_{i=1}^{k} (1 - \hat{y}_i) \). Note that \( 1 - \prod_{i=1}^{k} (1 - \hat{y}_i) \) is minimized when all the \( \hat{y}_i \)'s are equal (by symmetry). Namely, when \( \hat{y}_i = \tilde{z}_j/k \). Consider the function \( f(x) = 1 - (1 - x/k)^k \). This function is larger than \( g(x) = \beta_k x \), for all \( 0 \leq x \leq 1 \), as can be easily verified (see Tedium 47.2.3).

Thus,

\[
\mathbb{P}[C_j \text{ is satisfied}] = 1 - \prod_{i=1}^{k} (1 - \hat{y}_i) \geq f(\tilde{z}_j) \geq \beta_k \tilde{z}_j.
\]

The second part of the inequality, follows from the fact that \( \beta_k \geq 1 - 1/e \), for all \( k \geq 0 \). Indeed, for \( k = 1, 2 \) the claim trivially holds. Furthermore,

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

but this holds since \( 1 - x \leq e^{-x} \) implies that \( 1 - \frac{1}{k} \leq e^{-1/k} \), and as such \( (1 - \frac{1}{k})^k \leq e^{-k/k} = 1/e \). \( \square \)

**Tedium 47.2.3.** Consider the two functions

\[
f(x) = 1 - (1 - x/k)^k \quad \text{and} \quad g(x) = \left(1 - \left(1 - \frac{1}{k}\right)^k\right) x.
\]

We have \( f'(x) = (1 - x/k)^{k-1} \) and \( f''(x) = -k(k-1)/(1 - x/k)^{k-2} \). That is \( f''(x) \leq 0 \), for \( x \in [0, 1] \). As such \( f \) is a concave function.

Observe that \( f(0) = 0 = g(0) \) and \( f(1) = \left(1 - (1 - \frac{1}{k})^k\right) = g(1) \). Since \( f \) is concave, and \( g \) is linear, it follows that \( f(x) \geq g(x) \), for all \( x \in [0, 1] \).

**Theorem 47.2.4.** Given an instance \( I \) of MAX-SAT, the expected number of clauses satisfied by linear programming and randomized rounding is at least \( (1 - 1/e) \approx 0.632 \) \( m_{\text{opt}}(I) \), where \( m_{\text{opt}}(I) \) is the maximum number of clauses that can be satisfied on that instance.
Theorem 47.2.5. Given an instance \( I \) of MAX-SAT, let \( n_1 \) be the expected number of clauses satisfied by randomized assignment, and let \( n_2 \) be the expected number of clauses satisfied by linear programming followed by randomized rounding. Then, \( \max(n_1, n_2) \geq (3/4) \sum_j \widehat{z}_j \geq (3/4)m_{\text{opt}}(I) \).

Proof: It is enough to show that \( (n_1 + n_2)/2 \geq \frac{3}{4} \sum_j \widehat{z}_j \). Let \( S_k \) denote the set of clauses that contain \( k \) literals. We know that

\[
n_1 = \sum_k \sum_{C_j \in S_k} \left( 1 - 2^{-k} \right) \geq \sum_k \sum_{C_j \in S_k} \left( 1 - 2^{-k} \right) \widehat{z}_j.
\]

By Lemma 47.2.2 we have \( n_2 \geq \sum_k \sum_{C_j \in S_k} \beta_k \widehat{z}_j \). Thus,

\[
\frac{n_1 + n_2}{2} \geq \sum_k \sum_{C_j \in S_k} \frac{1 - 2^{-k} + \beta_k}{2} \widehat{z}_j.
\]

One can verify that \( (1 - 2^{-k}) + \beta_k \geq 3/2 \), for all \( k \). \(^2\) Thus, we have

\[
\frac{n_1 + n_2}{2} \geq \frac{3}{4} \sum_k \sum_{C_j \in S_k} \widehat{z}_j = \frac{3}{4} \sum_j \widehat{z}_j.
\]

47.3. From previous lectures

Theorem 47.3.1. Let \( X_1, \ldots, X_n \) be \( n \) independent random variables, such that \( \mathbb{P}[X_i = 1] = \mathbb{P}[X_i = -1] = \frac{1}{2} \), for \( i = 1, \ldots, n \). Let \( Y = \sum_{i=1}^{n} X_i \). Then, for any \( \Delta > 0 \), we have

\[
\mathbb{P}[Y \geq \Delta] \leq \exp(-\Delta^2/2n).
\]

References


\(^2\) Indeed, by the proof of Lemma 47.2.2, we have that \( \beta_k \geq 1 - 1/e \). Thus, \( (1 - 2^{-k}) + \beta_k \geq 2 - 1/e - 2^{-k} \geq 3/2 \) for \( k \geq 3 \). Thus, we only need to check the inequality for \( k = 1 \) and \( k = 2 \), which can be done directly.