13. Maximum Satisfiability

Satisfiability problems essentially involve solving CNF formulas

$$(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \bar{x}_2 \vee x_5) \wedge \dots$$

MAXSAT: Given a set of m clauses over n-variables, determine the max number of clauses that can be satisfied.

Randomized Algorithm to solve MAXSAT

We set each variable x_i to T or F with equal probability

Theorem - Given any m clauses, there is a truth assignment for the variables that satisfies at least $\frac{m}{2}$ clauses.
Proof -
We define Z_i, $1 \leq i \leq m$ which indicates if the clause has been satisfied or not.

$$Z_i = \begin{cases} 1 & \text{if clause } i \text{ is satisfied} \\ 0 & \text{otherwise} \end{cases}$$

Given that we randomly assign each variable x_i a value,

$$\mathbb{P}[Z_i = 1] = 1 - 2^{-k}$$

Given there are k literals in clause i (there's only 1 assignment of the k literals which would result in $Z_i = 0$).

Since we know $k \geq 1$, we know $\mathbb{P}[Z_i = 1] \geq \frac{1}{2}$.

Calculating expected number of satisfied clauses,

$$\mathbb{E}\left[\sum_{i=1}^{m} Z_i\right] = \sum_{i=1}^{m} \mathbb{E}[Z_i] \geq \frac{m}{2}$$

Now, we find the probability that the actual number of clauses is less than the expected value. Using a technique called *reverse markov*.

$$\mathbb{P}[Z < \mathbb{E}[Z]] = \mathbb{P}[m - Z > m - \mathbb{E}[Z]] < \frac{\mathbb{E}[m - Z]}{m - \mathbb{E}[Z]} = 1$$

Hence, there's a non-zero chance that the actual number of clauses satisfied is greater than the expected value, directly implying

$$\mathbb{P}\left[Z \geq \frac{m}{2}\right] \neq 0$$

Hence proved.

This type of proof is known as **Probabilistic Method**, where you show the existence of an object by showing it has a non-zero probability of occurring when chosen randomly.

Performance Ratio

Given an instance I, let $m_*(I)$ be the maximum number of clauses that can be satisfied. Let $m_A(I)$ be the number of clauses satisfiable by algorithm A.

We define the performance ratio of A to be definedas follows

$$\text{PerfRatio(A)} = \inf_I \frac{m_A(I)}{m_*(I)}$$

If $\text{PerfRatio(A)} = \alpha$, then A is an $\alpha-$approximation algorithm. The randomized algorithm discusses above is a $\frac{1}{2}$-approximation algorithm.

The approximation factor of our algorithm can be improved if we are allowed to assume that every clause has at least 2 literals, giving us a $\frac{3}{4}$-approximation algorithm.

LP Relaxations & Randomized Rounding

Forming an Integer Program for MAXSAT

We define variable $z_j \; \forall j \in [m]$

$$z_j = \begin{cases} 1 & \text{if clause } j \text{ is satisfied} \\ 0 & \text{otherwise} \end{cases}$$

define variable $y_i \; \forall i \in [n]$

$$y_i = \begin{cases} 1 & \text{if } x_i \text{ is True} \\ 0 & \text{otherwise} \end{cases}$$

and C_j^+ as the set of literals that appear unnegated in clause j, while C_j^- is the set of literals that appear negated in clause j.

We want to maximize

$$\sum_{j=1}^{m} z_j$$

Subject to constraints

1. y_i and $z_j \in \{0,1\}\ \forall i \in [n], \forall j \in [m]$.
2. $\sum_{i \in C_j^+} y_i + \sum_{i' \in C_j^-} (1 - y_{i'}) \geq z_j\ \forall j \in [m]$

Relaxing Integer Constraints

Solving an Integer Programming problem can take exponential time, as opposed to LP, which can take polynomial time, hence we relax the integer constraints on y_i, z_j, updating the first constraint to

1. y_i and $z_j \in [0,1]\ \forall i \in [n], \forall j \in [m]$

The LP will then provide us the solutions $\hat{y}_i, \hat{z}_j$ such that $\sum_{j=1}^{m} z_j \leq \sum_{j=1}^{m} \hat{z}_j$ (the LP has essentially more solutions to work with, so it's best case will be greater than or equal to the integer constraints case).

Randomized Rounding

Given $\hat{y}_i, \hat{z}_j$, we perform randomized rounding on them to obtain y_i, z_j

$$y_i = \begin{cases} 1 & \text{w.p } \hat{y}_i \\ 0 & \text{otherwise} \end{cases}$$

$$z_j = \begin{cases} 1 & \text{w.p } \hat{z}_j \\ 0 & \text{otherwise} \end{cases}$$

Lemma - Let clause C_j have k literals, The probability that C_j is satisfied by randomized rounding is $\geq \beta_k \hat{z}_j$ where $\beta_k = 1 - (1 - \frac{1}{k})^k$
Proof -
Since we are working with one clause, we just assume nothing is negated for

simplicity, i.e

$$C_j = x_1 \vee x_2 \vee \cdots \vee x_k$$

From the linear program, we know that $\sum\limits_{i=1}^{k} \hat{y}_i \geq \hat{z}_j$

For the clause to not be satisfied, all $\hat{y}_i$ should've been rounded down to 0, and we need to compute the probability of that not occurring.

$$\begin{aligned}
\mathbb{P}[C_j \text{ is satisfied}] &= 1 - \prod_{i=1}^{k}(1 - \hat{y}_i) \\
&\geq 1 - \left[\frac{\sum\limits_{i=1}^{k}(1 - \hat{y}_i)}{k}\right]^k && \text{(AM-GM Inequality)} \\
&\geq 1 - \left[\frac{k - \hat{z}_j}{k}\right]^k && \left(\sum_{i=1}^{k} \hat{y}_i \geq z_j\right) \\
&= 1 - \left(1 - \frac{\hat{z}_j}{k}\right)^k \\
&\geq \left[1 - \left(1 - \frac{1}{k}\right)^k\right]\hat{z}_j \\
&= \beta_k \hat{z}_j
\end{aligned}$$

The last step makes use of the claim that when $f(x) = 1 - (1 - \frac{x}{k})^k$ and $g(x) = \beta x$, then $f(x) \geq g(x)\ \forall x \in [0, 1]$.