Best-SAT

YouTube

Tomáš Sláma
20. 5. 2022

This post is an expanded translation of my lecture notes from a Randomized and Approximation Algorithms course that I took, and a more detailed explanation of the topics covered in my **video about BEST-SAT.**

Basic definitions

Definition (Optimalization problem) is a tuple $\mathcal{I}, \mathcal{F}, f, g$

- set of all **input instances** $\mathcal{I}$ [1]
- sets of **permissible inputs** $\forall I \in \mathcal{I} : \mathcal{F}(I)$
- **utility function** $\forall I \in \mathcal{I}, A \in \mathcal{F}(I) : f(I, A)$
- whether we're **maximizing** or **minimizing** (a single bit g)

Definition (NP-Optimalization problem) is an optimalization problem $\mathcal{I}, \mathcal{F}, f, g$, for which we additionally require that:

- the length of all permissible solutions is polynomial
- the language of $(I, A), I \in \mathcal{I}, A \in \mathcal{F}(I)$ is polynomial
  - *we can check the correctness of a solution in polynomial time*
- f is computable in polynomial time

Definition: algorithm A is R-approximation, if:

- it computes the solution in polynomial time (in terms of $|I|$) [2]
- for minimalization problem: $\forall I : f(A) \leq R \cdot \mathrm{OPT}(I)$
- for maximalization problem: $\forall I : f(A) \geq \mathrm{OPT}(I)/R$

MAX-SAT

- *Input:* $C_1 \wedge \ldots \wedge C_n$, each clause is a disjunction of $k_j \geq 1$ literals
- *Output:* evaluation $a \in \{0,1\}^n$ of the variables (sometimes called literals)
- *Goal:* maximize the number of satisfied clauses $\sum w_j$

We also assume that:

- no literal repeats in a clause
- at most one of $x_i, \overline{x}_i$ appearas in a clause

RAND-SAT

Algorithm (RAND-SAT)

1. choose all literals randomly (independently, for $p = 1/2$)
2. profit?

Theorem: RAND-SAT is a 2-approximation algorithm.

Proof: we'll create an indicator variable Y_j for each clause

- the chance that C_j is not satisfied is $\frac{1}{2^k}$

Since the size of the clause $k \geq 1$, we get $\mathbb{E}\left[Y_j\right] = \Pr\left[C_j \text{ is satistied}\right] = 1 - \frac{1}{2^k} \geq \frac{1}{2}$, thus

$$\mathbb{E}\left[\sum_{j=1}^{n} Y_j\right] \overset{\substack{\text{linearity} \\ \text{of expectation}}}{=} \sum_{j=1}^{n} \mathbb{E}\left[Y_j\right] \geq \sum_{j=1}^{n} \frac{1}{2} \geq \frac{1}{2}\mathrm{OPT}$$

[1] An example problem could be minimum spanning trees:

- **input instances:** *set of all weighted graphs*
- **permissible inputs:** *spanning trees for the given weighted graph*
- **utility function:** *the spanning tree weight (sum of its edges)*
- we're **minimizing**

[2] *For minimalization problem, we ensure that the solution is always small enough. For maximalization problem, we ensure that the solution is always large enough.*

LP-SAT

Algorithm (LP-SAT) [3]

1. build an integer linear program:
   - variables will be:
     - y_i for *each literal*
     - z_j for *each clause*
   - inequalitites will be one for *each clause*, in the form

$$z_j \leq \sum_{\text{positive}} y_i + \sum_{\text{negative}} (1 - y_i)$$

   - we'll *maximize* the number of satisfied clauses $\sum z_j$
2. relax the program (allow real variables instead of integers) and calculate the optimum y^*, z^*
3. set literals x_i to 1 with probability y_i^*

Theorem: LP-SAT is a $\left(1 - \frac{1}{e}\right)$-approximation algorithm.

To prove this, we'll use a few lemmas/theorems that aren't difficult to prove, but aren't really interesting. I left links to (Wikipedia *and I don't feel bad about it*) articles with proofs for each, if you're interested.

Fact (A - A/G mean inequality)

$$\prod_{i=1}^{n} a_i^{\frac{1}{n}} \leq \frac{1}{n} \sum_{i=1}^{n} a_i$$

Proof: https://en.wikipedia.org/wiki/Inequality_of_arithmetic_and_geometric_means

Fact (B - Jensen's inequality) if a function is concave on the interval $[0, 1]$ and $f(0) = a, f(1) = a + b$, then

$$\forall x \in [0, 1] : f(x) \geq a + bx$$

Proof: https://en.wikipedia.org/wiki/Jensen%27s_inequality

Fact (C - 1/e inequality)

$$\left(1 - \frac{1}{n}\right)^n \leq \frac{1}{e}$$

Proof: https://en.wikipedia.org/wiki/E_(mathematical_constant)#Inequalities

---

Proof (of the main theorem) consider y^*, z^* and C_j with k_j literals; then

$$\begin{aligned}
\Pr\left[C_j \text{ is not satisfied}\right] &= \overbrace{\prod_{i:\ x_i \in C_j} (1 - y_i^*)}^{\text{positive}} \overbrace{\prod_{i:\ \overline{x}_i \in C_j} y_i^*}^{\text{negative}} \\
&\overset{A}{\leq} \left[\frac{1}{k_j}\left(\sum_{i:\ x_i \in C_j} (1 - y_i^*) + \sum_{i:\ \overline{x}_i \in C_j} y_i^*\right)\right]^{k_j} \\
&= \left[1 - \frac{1}{k_j}\left(\sum_{i:\ x_i \in C_j} y_i^* + \sum_{i:\ \overline{x}_i \in C_j} (1 - y_i^*)\right)\right]^{k_j} \\
&\leq \left(1 - \frac{z_j^*}{k_j}\right)^{k_j}
\end{aligned}$$

---

[3] We're using the optimal solution to the linear program (and generally the formula, if we allow real vlaues for literals) as a guide for our randomized algorithm.

We're interested in the satisfied ones, so

$$\begin{aligned}
\Pr\left[C_j \text{ is satisfied}\right] &\geq \overbrace{1 - \left(1 - \frac{z_j^*}{k_j}\right)^{k_j}}^{\text{our function } f(z_j^*)} \\
&\overset{B}{\geq} \left[1 - \left(1 - \frac{1}{k_j}\right)^{k_j}\right] z_j^* \overset{C}{\geq} \left(1 - \frac{1}{e}\right) z_j^*
\end{aligned}$$

To use fact B, we observed that $a = f(0) = 0$ and that the second derivation is non-positive (so the function is concave). Now to formally count how many our program satisfies:

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

BEST-SAT

Algorithm (BEST-SAT)

1. assign a value of a literal using RAND-SAT with probability 1/2, else use BEST-SAT
2. have an existential crisis about the fact that this works and is asymptotically optimal

Theorem: BEST-SAT is $\frac{3}{4}$-approximation.

Proof: we want to prove that $\Pr\left[C_j \text{ is satisfied}\right] \geq \frac{3}{4} z_j^*$.

Let's look at the probability that each algorithm satisfies a clause of k variables:

- RAND-SAT: $1 - \frac{1}{2^k}$ (at least one literal must be satisfied)
- LP-SAT: $\left[1 - \left(1 - \frac{1}{k}\right)^k\right] z_j^*$ (the formula right before using fact C)

Now the proof boils down to the following table:

k_j	RAND-SAT	LP-SAT	BEST-SAT
1	$\frac{1}{2} \geq \frac{1}{2} z_j^*$	$1 \cdot z_j^*$	$\frac{1}{2}\frac{1}{2} + \frac{1}{2} z_j^* \geq \frac{3}{4} z_j^*$
2	$\geq \frac{3}{4} z_j^*$	$\frac{3}{4} \cdot z_j^*$	$\geq \frac{3}{4} z_j^*$
≥ 3	$\geq \frac{7}{8} z_j^*$	$\geq \left(1 - \frac{1}{e}\right) \cdot z_j^*$	$> \frac{3}{4} z_j^*$