

$\frac{3}{4}$ -approximation for Max Sat via Randomized Rounding ¹

- Recall the MAX SAT problem : the input is a CNF formula ϕ on n variables x_1, \dots, x_n and m clauses. Let \mathcal{C} denote the set of all clauses. Each clause is an AND of a collection of 1 or more literals. In particular, different literals could have a different number of literals, and some could even have just one. In this sense, this problem differs from MAX 2SAT where every clause had *exactly* two literals. The objective is to find an assignment of variables to $\{\text{true}, \text{false}\}$ so as to maximize the number of satisfied clauses. In this note, we describe a randomized $\frac{3}{4}$ -approximation.

Exercise: 🙋🙋🙋 Recall that we described a local search algorithm which was a $\frac{3}{4}$ -approximation for MAX 2SAT. Can the same algorithm be modified to give a $\frac{3}{4}$ -approximation for MAX SAT?

- The final $\frac{3}{4}$ -approximation will in fact be the “best of two” different randomized algorithms, each of which are individually are approximation algorithms with worse factors. The first algorithm is a very simple algorithm : independently assign each variable $x_i \in \{\text{true}, \text{false}\}$ uniformly at random.

1: **procedure** NAIVE RANDOMIZED(ϕ):
 2: Independently, assign each variable $x_i \in \{\text{true}, \text{false}\}$ uar.

Theorem 1. If every clause has at least k literals, NAIVE RANDOMIZED is an $(1 - 2^{-k})$ -approximation. In particular, it is a $\frac{1}{2}$ -approximation for MAX SAT.

Proof. Given a clause C , let Z_C be the indicator variable it is satisfied; $\mathbf{Exp}[\text{alg}] = \sum_{C \in \mathcal{C}} \mathbf{Exp}[Z_C]$. If the clause C has k literals, then $\mathbf{Pr}[Z_C = 1] = 1 - 2^{-k}$. \square

- An LP Relaxation and a $(1 - \frac{1}{e})$ -approximation. Next we see a “max-coverage” style $1 - 1/e$ -approximation via randomized rounding. We have a variable y_i for every variable x_i indicating whether it is set to true or false. For each clause C , we have a variable z_C indicating whether it is satisfied. Let C^+ be the variables in C which appear in C as is, and C^- be those variables which appear in C as the complement. The following is an LP relaxation for Max-SAT.

$$\text{opt} \leq \text{lp}(\phi) := \text{maximize} \quad \sum_{C \in \mathcal{C}} z_C \quad (\text{MaxSAT-LP})$$

$$z_C \leq \sum_{i \in C^+} y_i + \sum_{j \in C^-} (1 - y_j) \quad \forall C \in \mathcal{C} \quad (1)$$

$$0 \leq z_C, y_i \leq 1, \quad \forall i \in [n], C \in \mathcal{C} \quad (2)$$

¹Lecture notes by Deeparnab Chakrabarty. Last modified : 17th Jan, 2022
 These have not gone through scrutiny and may contain errors. If you find any, or have any other comments, please email me at deeparnab@dartmouth.edu. Highly appreciated!

- 1: **procedure** IND. RANDOMIZED ROUNDING(ϕ):
- 2: Solve (MaxSAT-LP) to get y_i 's and z_C 's.
- 3: Independently, assign each variable x_i the value true with probability y_i .


Theorem 2. IND. RANDOMIZED ROUNDING is an $(1 - \frac{1}{e})$ -approximate algorithm.

Proof. Fix a clause C . It suffices to prove that it is satisfied with probability $\geq z_C \cdot (1 - 1/e)$. If so, then we would be done by linearity of expectation; the expected number of clauses satisfied is $\geq (1 - 1/e) \cdot \text{lp}(\phi)$.

Indeed, it is satisfied if one of the variables in C^+ is set to true or if one of the variables in C^- is set to false. So, the probability it is **not** satisfied is

$$\begin{aligned} \Pr[C \text{ not satisfied}] &= \prod_{i \in C^+} (1 - y_i) \cdot \prod_{j \in C^-} y_j \quad \leq \quad \prod_{i \in C^+} e^{-y_i} \cdot \prod_{j \in C^-} e^{-(1-y_j)} \\ &= e^{-(\sum_{i \in C^+} y_i + \sum_{j \in C^-} (1-y_j))} \quad \leq e^{-z_C} \end{aligned}$$

Therefore, the probability C is satisfied is $\geq 1 - e^{-z_C} \geq z_C(1 - 1/e)$. In the last inequality we have used the fact that the function $g(t) = 1 - e^{-t}$ is concave, and thus when $t \in [0, 1]$ we have $g(t) \geq tg(1) + (1 - t)g(0) = t(1 - 1/e)$. Since $z_C \leq 1$, the proof follows. \square

Exercise:  Strengthen the above analysis to show that if C has k literals, then the probability it is satisfied is $z_C \cdot (1 - (1 - 1/k)^k)$. Of particular importance is the case $k = 1, 2$. That is, if C has 1 literal, then the probability it is satisfied is z_C (this should be immediate), and if C has 2 literals, then the probability it is satisfied is $\geq \frac{3}{4} \cdot z_C$. The main idea is to not use our “favorite” inequality, namely $1 + t \leq e^t$ for all t , but rather use AM-GM.

- *Obtaining 3/4 by taking best of both.* To get better than 3/4, one has to in fact do the exercise above. For the time being, let us believe the statement of the exercise. Then one notices the following : if the size of C is large, then the chance of satisfying the clause is “higher” for the first algorithm and “lower” for the second. Indeed, when C has only one clause (the “worst” case for the first algo), the second algorithm satisfies it with no loss when compared against z_C . This immediately suggests the following : run both algorithms and pick the best. One nice way of incorporating “best of two algorithms” is to flip a coin and pick one, and argue about the average. If this average itself is good, then the best can be only better.

- 1: **procedure** BEST OF BOTH(ϕ):
- 2: Flip a fair coin.
- 3: If it is heads, run NAIVE RANDOMIZED(ϕ).
- 4: If it is tails, run IND. RANDOMIZED ROUNDING (ϕ).

Theorem 3. BEST OF BOTH(ϕ) is a $\frac{3}{4}$ -approximation algorithm.

Proof. Fix a clause C . We now argue that the probability it is satisfied is $\geq \frac{3}{4} \cdot z_C$. This would prove the theorem. Note, crucially, that this probability is over the coin toss in [Line 2](#), and the coin-tosses in the individual algorithms. Indeed, suppose p_C is the probability C is satisfied in NAIVE RANDOMIZED and suppose q_C is the probability that C is satisfied in IND. RANDOMIZED ROUNDING. Then, the probability C is satisfied in BEST OF BOTH is exactly $\frac{p_C+q_C}{2}$. Let's call this π_C .

Suppose C has k literals. Then, we know $p_C = 1 - 2^{-k} \geq z_C(1 - 2^{-k})$ since $z_C \leq 1$. We also know $q_C \geq z_C(1 - 1/e)$. Therefore, if $k \geq 3$, we get

$$\pi_C \geq \frac{1}{2} \cdot \left(\frac{7}{8}z_C + \left(1 - \frac{1}{e}\right)z_C \right) \underbrace{>}_{\text{using a calculator}} 0.75z_C$$

If $k = 1, 2$, then we use the exercise above. When $k = 1$, we get $\pi_C = \frac{z_C}{4} + \frac{z_C}{2} = 0.75z_C$. When $k = 2$, we get $\pi_C = \frac{3z_C}{8} + \frac{3z_C}{8} = 0.75z_C$. \square

Notes

Approximation algorithms for Max-SAT was first studied in the seminal paper [4] by Johnson which contained the NAIVE RANDOMIZED algorithm described above. It also contained another greedy algorithm achieving the same ratio. The first $\frac{3}{4}$ -approximation for Max-SAT is given in the paper [6] by Yannakakis. Instead of using the linear programming relaxation explicitly, the paper constructs network flow instances and then uses this to get the assignment. The presentation here is from the paper [3] by Goemans and Williamson. More recently, there have been other simpler $\frac{3}{4}$ -approximation algorithms; we refer the reader to the papers [2] by Buchbinder, Feldman, Naor, and Schwarz, and [5] by Poloczek, Schnitger, Williamson, and van Zuylen. All of these algorithms can be derandomized. The best approximation algorithms for Max SAT have been obtained by rounding semidefinite programming (SDP) relaxations. The current record holder are algorithms in the paper [1]; one of their algorithms is a provable ≈ 0.797 -approximation, while another gives a ≈ 0.8434 -factor based on an unproved conjecture about a function in high-dimensional geometry.

References

- [1] A. Avidor, I. Berkovitch, and U. Zwick. Improved approximation algorithms for max nae-sat and max sat. In *Proc., Workshop on Approximation and Online Algorithms (WAOA)*, pages 27–40. Springer, 2005.
- [2] N. Buchbinder, M. Feldman, J. S. Naor, and R. Schwartz. A tight linear time $(1/2)$ -approximation for unconstrained submodular maximization. *SIAM Journal on Computing (SICOMP)*, 44(5):1384–1402, 2015.
- [3] M. X. Goemans and D. P. Williamson. New $\frac{3}{4}$ -approximation algorithms for the maximum satisfiability problem. *SIAM Journal on Discrete Mathematics (SIDMA)*, 7(4):656–666, 1994.
- [4] D. S. Johnson. Approximation algorithms for combinatorial problems. *J. Comput. System Sci.*, 9(3):256–278, 1974.
- [5] M. Poloczek, G. Schnitger, D. P. Williamson, and A. Van Zuylen. Greedy algorithms for the maximum satisfiability problem: Simple algorithms and inapproximability bounds. *SIAM Journal on Computing (SICOMP)*, 46(3):1029–1061, 2017.
- [6] M. Yannakakis. On the approximation of maximum satisfiability. *J. Algorithms*, 17(3):475–502, 1994.