An algorithm for MAX2SAT

Cristian Dumitrescu

independent mathematician

Abstract- In this paper I present a MAX2SAT algorithm based on the randomized algorithm of Papadimitriou from 1991. We also show that this algorithm finds a MAX2SAT solution (if it exists) with high probability in polynomial time.

Index Terms- NP-complete problems, MAX2SAT, Markov chain, 3SAT

I. Introduction

In MAX2SAT we are given a 2SAT expression involving a set of clauses, each with at most two literals in it, and an integer K. We are asked whether there is a truth assignment that satisfied at least K of the clauses. We call this the MAX2SAT problem. We note that if the 2SAT expression considered has m clauses, then the truth assignment mentioned above must leave at most $m - K$ clauses unsatisfied.

Definition 1. We know that there is (at least) a truth assignment that satisfies a maximum number of clauses $M$and leaves unsatisfied a minimum number of clauses $m - M$, where m represents the number of clauses of the 2SAT expression under consideration. The truth assignment satisfying this condition will be called the maximal truth assignment (there is at least one).

Proposition 1. (see [5] and [3]). MAX2SAT is NP- complete.

Proof. For the proof, see the references listed above (see [5] , theorem 9.2, page 186).

II. Presentation of the algorithm

We will present the well known randomized algorithm for 2SAT. This is the Papadimitriou algorithm from 1991 (see [4]). We also note that Schoning discussed a similar algorithm for 3SAT in 1991 (see [6])

```
Papadimitriou's algorithm for 2SAT.

Input: a 2SAT expression in $n$ variables.

Guess an initial truth assignment, uniformly at random.

Repeat $C \cdot n^2$ times:

        If the formula is satisfied by the actual assignment, stop and accept.

        Let C be some clause not being satisfied by the actual assignment. Pick one of the (at most) two literals in the clause at
        random, and flip its truth value.

        Update.

Stop and reject, the expression is not satisfiable.
```

In the next section, I will prove that this algorithm performs well even for the MAX2SAT problem, so in the Papadimitriou algorithm, all we need to modify is the definition of “satisfies” (related to an assignment), since in the context of MAX2SAT, we will not need all the clauses to be satisfied.

```
The MAX2SAT algorithm.

Input: a 2SAT expression with $m$ clauses.

Guess an initial truth assignment, uniformly at random.

Repeat $C \cdot m^2$ times:

        If the actual truth assignment satisfies the MAX2SAT condition, stop and accept.

        Let C be some clause not being satisfied by the actual assignment. Pick one of the (at most) two literals in the clause at
        random, and flip its truth value.

        Update.

Stop and reject, the expression is not satisfiable.
```

In other words, I will prove that if a MAX2SAT satisfying assignment exists, then the MAX2SAT algorithm will find it with high probability. Note that running the algorithm a quadratic number of times, in the number of clauses is stronger than running it a quadratic number of times in the number of variables. A given 2SAT expression with m clauses can have at most $2m$ variables, and at least a number of variables of order of magnitude $\sqrt{m}$ (up to a multiplicative constant).

III. Analysis of the Algorithm

In this section, I will present the main theorem. We assume that the reader is also familiar with the 3SAT to MAX2SAT reduction. For reference, see [5] , theorem 9.2, page 186. A given 2SAT problem is called a reduction of a 3SAT problem if it is equivalent to a given 3SAT problem.

Theorem. Given a MAX2SAT problem, that is a reduction of a given 3SAT problem with m clauses, then with high probability, after $C \cdot m^2$ steps, the MAX2SAT algorithm will find a MAX2SAT satisfying assignment (where C is a constant and m is the number of clauses in the original 3SAT expression).

Proof. For the proof, we will use the principle of induction, following the number of clauses in the MAX2SAT expression (actually the number of clauses of the original 3SAT expression). We will consider the Markov chain approach considered in [6] and [4]. We will also consider the expected travelled distance (by the Markov chain associated to the algorithm) towards the absorbing barrier 0 (note that the direction of travel is important).

We are given a 3SAT problem with m clauses. We construct the equivalent MAX2SAT problem. The MAX2SAT problem will have $10 \cdot m$ clauses, each 3SAT clause will be replaced by a group of 10 clauses. The 3SAT problem is satisfiable if and only if its MAX2SAT reduction has a maximal truth assignment with $7 \cdot m$ satisfied clauses.

We write $P(N)$ for the statement : after $C \cdot N^2$ steps (where $10 \cdot N$ is the number of clauses in the MAX2SAT expression that is equivalent to a 3SAT expression with N clauses), the Markov chain associated to the MAX2SAT algorithm will have an expected travel distance (towards the absorbing state with Hamming distance 0) given by the square root of the number of flips performed by the algorithm (up to a multiplicative constant), on any MAX2SAT expression that is equivalent to a 3SAT expression with N clauses. So if the algorithm performs $C \cdot N^2$ steps, then the expected travel distance is $C_1 \cdot \sqrt{C} \cdot N$ (where C_1 is a constant greater than 1).

For $N = 1$, the original 3SAT expression has only one clause. By choosing the constant C large enough, it is clear that the statement P(1) is true.

We want to prove that if we assume that $P(\alpha)$ is true for all integers $\alpha < N$, then $P(N)$ is also true.

Let's consider a 2SAT expression φ with $10 \cdot N$ clauses (the MAX2SAT equivalent to a 3SAT expression with N clauses). We consider the expression ψ , by eliminating a group of 10 clauses (associated to one of the clauses of the original 3SAT expression). We call this group, the special group of 10 clauses. The expression ψ will have $10 \cdot (N - 1)$ clauses. When working on the expression φ , we assume that the algorithm will touch this special group of clauses a number of times K. That means that if the algorithm takes $C \cdot N^2$ steps (where N is the number of clauses in the expression φ), then $C \cdot N^2 - K$ steps will be taken within the expression ψ , and K steps will involve the special group of clauses considered above.

We assume that the associated Markov chain will take n_1 steps within the expression ψ, then it will touch the special group of clauses considered above, then it will take n_2 steps within the expression ψ, then it will touch the special group of clauses considered above again, and so on, until it will take n_K steps within the expression ψ, then it will touch the special group of clauses considered above for the last time, and then it will take the last n_{K+1} steps within the expression ψ.

We are only interested in studying what happens if the original 3SAT expression is satisfiable.

The special group of clauses will have 3 unsatisfied clauses in the maximal truth assignment. The whole expression φ will have $3 \cdot N$ unsatisfied clauses in the maximal truth assignment. As long as the algorithm does not find a maximal truth assignment, the probability of choosing an unsatisfied clause from the special group is at most $p = \frac{3}{3N} = \frac{1}{N}$. That means that if the algorithm makes $C \cdot N^2$steps, then the expected number of hits of an unsatisfied clause from the special group is at most $C \cdot N$. In other words, the inequality $K < C \cdot N$ will be satisfied with high probability.

From the inductive hypothesis, when the algorithm works within ψ , it will drift towards zero with an expected speed, and when it touches an unsatisfied clause in the special group of clauses, the worst it can do is to backtrack one unit (away from the absorbing state 0).

That means that we have the following equation (this counts the number of steps within ψ , and when it touches the unsatisfied clause considered above).

$$n_1 + n_2 + n_3 + \cdots \ldots \ldots \ldots + n_{K+1} = C \cdot N^2 - K \qquad (1)$$

The overall, expected travel distance of the associated Markov chain (when the algorithm works on φ), will be at least:

$$D = C_1 \cdot ((n_1)^{\frac{1}{2}} + (n_2)^{\frac{1}{2}} + \cdots \ldots \ldots . + (n_{K+1})^{\frac{1}{2}}) - K \qquad (2)$$

We also work under the conditions $n_i \geq 1$, for all i $\in \{1, 2, 3, \ldots . K+1\}$, and we assume that $K < C \cdot N$, as we discussed before.

We want to find the minimum value that D can take, under the constraint given by equation (1). We can apply the method of Lagrange multipliers, more precisely the Karush-Kuhn-Tucker minimization conditions, and we find that the minimum of D has the form $C_2 \cdot N$, where the constant C_2 is greater than 1 (for a suitably chosen constant C).

There is also a geometrical way to see this. We consider the variables
$z_i = n_i^{\frac{1}{2}}$, for all i $\in \{1, 2, 3, \ldots . K+1\}$. We have to study the intersection of a hyperplane and a hypersphere (a K - sphere), in the region where all the coordinates are greater or equal to 1. The hyperplane is given by the equation
$C_1 \cdot (z_1 + z_2 + \cdots \ldots + z_{K+1}) =$ constant , and the hypersphere is given by the equation $z_1{}^2 + z_2{}^2 + z_3{}^2 + \cdots \ldots . . z_{K+1}{}^2 = C \cdot N^2 - K$. When we perform a translation of the hyperplane, parallel to itself, eventually will hit the region where all the coordinates are greater or equal to 1 (corresponding to a slice of the hypersphere that is less than $\frac{1}{2^{K+1}}$ of the whole).

For any translation of the hyperplane further, the sum $(z_1 + z_2 + \cdots \ldots + z_{K+1})$ will only increase. Lets' see the values at the point $z_1 = \sqrt{C \cdot N^2 - 2 \cdot K}$, $z_2 = z_3 = \ldots . = z_{K+1} = 1$. We look at the equations from a strictly formal point of view (disregarding for a moment the interpretation of these values). In this case, the distance D will take the value:

$$D = C_1 \cdot \sqrt{C \cdot N^2 - 2 \cdot K} + C_1 \cdot K - K \qquad (3)$$

Knowing that with high probability we have $K < C \cdot N$, from (3) we see that the drift distance D can be put in the form (when K hits its maximum value)

$$D \sim (C_1 \cdot (\sqrt{C} + 1) - \sqrt{C}) \cdot \sqrt{C} \cdot N . \qquad (4)$$

If $C_1 > 1$, then we have $C_2 = (C_1 \cdot (\sqrt{C} + 1) - \sqrt{C}) > 1$

We emphasize that this is only an approximation, the KKT minimization method (mentioned above) will give the exact minimum.

That means that P(N)is true, and the inductive steps is proved. QED.

The MAX2SAT algorithm always finds a maximal truth assignment, not only when the maximal truth assignment satisfies all the clauses

When a literal is flipped in an unsatisfied clause that will be satisfied in the maximal truth assignment, the Hamming distance to a solution decreases by 1 with probability at least$\frac{1}{2}$. When a literal is flipped in an unsatisfied clause that will remain unsatisfied in the maximal truth assignment, the Hamming distance indeed increases by 1, but enough clauses will become unsatisfied at this step, with this flip (and these clauses will be satisfied in the maximal truth assignment) , such that the deviation will be corrected al later stages of the algorithm. In general terms, this could be one reason why the MAX2SAT algorithm works in all cases, as the proof of the theorem above shows (using induction). A direct proof of the theorem above (without induction) would not be a simple matter. The general procedure will look like this. We consider a 3SAT problem. We find the equivalent MAX2SAT problem, and we run the MAX2SAT algorithm in this problem

IV. CONCLUSION

For general implications, related to efficiently solving NP – complete problems, see [1]. An interesting application is related to the problem of automated theorem proving using an efficient algorithm for NP – complete problems (see appendix]). The impact of this type of algorithm in mathematics, cryptography, science in general is hard to estimate.

APPENDIX

Godel's letter to John von Neumann. In his letter, Godel writes:
"One can obviously easily construct a Turing machine, which for every formula F in first order predicate logic and every natural number n, allows one to decide if there is a proof of F of length n (length = number of symbols). Let $\psi(F,n)$ be the number of steps the machine requires for this and let $\varphi(n) = max_F \psi(F,n)$. The question is how fast $\varphi(n)$ grows for any optimal machine" (see [2].

Now we consider this. In [3], we have theorem 13.1, at page 325, where it is proved that for each Turing machine (deterministic or nondeterministic) M that is time bounded by a polynomial $p(n)$, a log-space algorithm exists, that takes as input a string x and produces a Boolean expression E_x that is satisfiable if and only if M accepts E_x.

This means that the process of seeking a proof (of reasonable length) of a mathematical statement can be completely automatized. With the algorithm presented in this paper, Godel's vision can be made reality.

ACKNOWLEDGMENT

I emphasize that Professor Fortnow's book ([1]) was a great source of inspiration and motivation that led me to seriously consider this problem. Also, Professor Uwe Schoning , Professor Lance Fortnow, and Professor C. Papadimitriou corrected some errors that I had in many previous versions of this paper, and for that I am very grateful. I must emphasize though, that the final version of my paper did not receive feedback from these experts in the field, so I do not claim that I have their endorsement. If there are still flaws in my work, that is all my fault

REFERENCES

[1]]. L. Fortnow, " *The Golden Ticket, P, NP, and The Search For The Impossible* ", Princeton University Press, 2013.
[2] K. Godel, " *Kurt Godel's letter to John von Neumann*", Princeton, 20 march 1956, http://www.cs.cmu.edu/~aada/courses/15251s15/www/notes/godel-letter.pdf.
[3] J. E Hopcroft, J. D. Ullman" *Introduction to Automata Theory, Languages and Computation* ",Addison - Wesley Publishing Company, Inc., 1979.
[4] C.H. Papadimitriou, "*On selecting a satisfying truth assignment* ", in FOCS, pages 163 – 169, 1991.
[5] C.H. Papadimitriou, "*Computational Complexity* ", Addison - Wesley Publishing Company, Inc., 1994.
[6] Uwe Schoning, " *A probabilistic Algorithm for k-SAT and Constraint Satifaction Problems*", Research Supported by the ESPRIT Basic Research, 1991.

AUTHORS

Cristian Dumitrescu – independent mathematician, BSc. In Mathematics, cristiand43@gmail.com

Correspondence Author – Cristian Dumitrescu, cristiand43@gmail.com , 1-(519)-574-7026