CSE 203A Lecture 4
Randomized Algorithms for Approximating MAXSAT

Ramamohan Paturi

January 21, 2014

Scribe Notes by: Jiawei Gao

1 Problem Description

SAT

In the SAT problem, we are given a CNF (conjunctive formal form) formula F consisting of n Boolean variables $x_1, \ldots, x_n$. It has m clauses $C_1, \ldots, C_m$. Each clause is a disjunction of variables and their negations, which we call literals. The CNF is a conjunction of all the clauses. The problem SAT (satisfiability) is to decide whether the CNF is satisfiable. If F is a k-CNF (each clause contains at most k literals), we say it is a k-SAT problem. For $k \geq 3$, there are no efficient algorithms. The best known algorithm for k-SAT is 2^n times the size of the formula. Furthermore, the Exponential Time Hypothesis (ETH) assumes that 3-SAT cannot be solved in subexponential time.

MAXSAT

In MAXSAT problem we want to find an assignment to satisfy the maximum number of clauses. Variations of MAXSAT include weighted MAXSAT, where each clause C_i as associated with a weight $W_i \geq 0$, and we are to find a set of satisfied clauses to maximize the sum of weight. In this note we only look at non-weighted MAXSAT. The argument for weighted MAXSAT is almost the same.

We assume that each clause has exactly k literals, and no repeated variables appear in the same clause. Also, assume that there are no identical clauses.

With probabilistic algorithms we can find efficient approximation algorithms for MAXSAT. For $F(x_1, \ldots x_n) = \bigwedge_{i=1}^{m} C_i$, we wish to satisfy $l \geq \frac{1}{2}m$ clauses in F.

2 Markov's Inequality

Our algorithm independently assign all the variables with random values. Let $x_1, \ldots x_n$ be n independently and uniformly distributed random variables. Let $\#F(x_1, \ldots x_n)$ denote the number of satisfied clauses in F.

We use m Boolean variables $Z_i \in \{0, 1\}$ for $1 \leq i \leq m$ to indicate the Boolean value of clauses. Define Z_i as

$$Z_i = \begin{cases} 1 & \text{if } C_i \text{ is true;} \\ 0 & \text{otherwise.} \end{cases}$$

Then $\#F = \sum_{i=1}^{m} Z_i$.

Theorem 2.1 (linearity of expectations)**.** *For any finite collection of discrete random variables $X_1, \dots X_n$ with finite expectations,*

$$E\left[\sum_{i=1}^{n} X_i\right] = \sum_{i=1}^{n} E[X_i].$$

Proof. First, for two variables, we prove $E[X+Y] = E[X] + E[Y]$.

$$\begin{aligned}
E[X+Y] &= \sum_i \sum_j (i+j) \cdot Pr((X = i \cap Y = j) \\
&= \sum_i \sum_j i \cdot Pr((X = i \cap Y = j) + \sum_i \sum_j j \cdot Pr((X = i \cap Y = j) \\
&= \sum_i i \sum_j Pr((X = i \cap Y = j) + \sum_j j \sum_i Pr((X = i \cap Y = j) \\
&= \sum_i i \cdot Pr(X = i) + \sum_j j \cdot Pr(Y = j) \\
&= E[X] + E[Y].
\end{aligned}$$

By induction the theorem holds for n variables. □

By linearty of expectation,

$$E[\#F] = E\left[\sum_{i=1}^{m} Z_i\right] = \sum_{i=1}^{m} E[Z_i]$$

.

For each clause, the probability of it being satisfied is the probability of any of its literals set to true. A literal is true by probability 1/2, and different literals are independent. Thus the clause is not satisfied by probability 2^{-k}. Thus $Pr(Z_i = 1) = 1 - 2^{-k}$.

$$\begin{aligned}
E[Z_i] &= Pr(Z_i = 1) \cdot 1 + Pr(Z_i = 0) \cdot 0 \\
&= (1 - 2^{-k}) \cdot 1 + 2^{-k} \cdot 0 \\
&= 1 - 2^{-k}
\end{aligned}$$

When k is at least 1, $E[\#F] = \sum_{i=1}^{m} (1 - 2^{-k_i}) \geq \frac{1}{2}m$; When $k \geq 3$, the expected value $E[\#F] \geq 1 - 2^{-3} = \frac{7}{8}m$. This is one of the fundamental results in approximation theory.

Algorithm 1 MAXSAT approximation algorithm

```
1: function MAXSAT
2:     loop for many times
3:         Randomly assign all variables and count the satisfied clauses
4:     end loop
5:     return the maximum #F we have found in all the trials
6: end function
```

Next we compute the probability of our algorithm finding a $\#F$ that is at least half of the number of clauses. Then we compute how many times we need to repeat the algorithm in order to make it high probability to find $m/2$ satisfied clauses.

Definition 2.1 (High probability)**.** If $Pr(A) \geq 1 - \frac{1}{n^k}$, we say event A occurs with high probability.

To achieve a high probability of satisfying $m/2$ clauses, how many times do we need to try? We'll compute the probability of success using Markov's inequality.

Theorem 2.2 (Markov's inequality)**.** *Let X be a random variable that assumes only nonnegative values. Then, for all $a > 0$,*

$$Pr(X \geq a) \leq \frac{E[X]}{a}$$

.

Proof. Let

$$I = \begin{cases} 1 & \text{if } X \geq a; \\ 0 & \text{otherwise.} \end{cases}$$

Then $I \leq X/a$. $E[I] \leq E[X/a] = E[X]/a$. And because $E[I] = Pr(I = 1) = Pr(X \geq a)$, thus $Pr(X \geq a) \leq E[X]/a$. □

In MAXSAT, we use the same technique. Let W denote the event $(\#F \geq \frac{1}{2}m)$ and let $q = Pr(W)$. Then for $Pr(\#F < \frac{1}{2}m) = 1 - q$. Because the number of clauses are integers, $\#F < \frac{1}{2}m$ is equal to $\#F \leq \frac{1}{2}m - 1$.

Because $\#F$ is the number of satisfied clauses, by definition $\#F \leq m$. Thus with probability $1 - q$, we have the bound $\#F \leq \frac{1}{2}m - 1$; And in the complementary event with probability q, the bound is $\#F \leq m$. Then we have $E(\#F) \leq (1 - q)(\frac{1}{2}m - 1) + qm, \Rightarrow q \geq \frac{1}{m}$.

From the argument above, the probability of failure is $1 - q \leq 1 - \frac{1}{m} \leq e^{-1/m}$.

By Repeating m times, the probability of failure becomes $(1 - \frac{1}{m})^m \to \frac{1}{e}$; By repeating $m \lg n$ times, the probability of failure is $1/n$, making it high probability to success.

3 Derandomization using k-wise Independence

Do we actually need to try 2^n assignments? Notice that in k-SAT problem, we only use the fact that the k variables in the same clause are independent. Thus the variables only need to be k-wise independent. In this way we can significantly reduce the sample space.

Definition 3.1 (k-wise independence)**.** Random variables $X_1, \ldots X_n$, are k-wise independent if

$$\forall X_{i_1}, \ldots X_{i_k}, \forall b_1, \ldots b_k, Pr(X_{i_1} = b_1, \ldots X_{i_k} = b_k) = \frac{1}{2^k}.$$

Example: $n = 4, k = 2$,

x_1	x_2	x_3	x_4	Probability
0	0	0	0	1/8
0	1	0	1	1/8
0	0	1	1	1/8
0	1	1	0	1/8
1	1	1	1	1/8
1	0	1	0	1/8
1	1	0	0	1/8
1	0	0	1	1/8

The four variables are not independent. But if we take any pair of the variables, say x_1 and x_3, then it is easy to check that

$$Pr(x_1 = 0, x_3 = 0) = Pr(x_1 = 0, x_3 = 1) = Pr(x_1 = 1, x_3 = 0) = Pr(x_1 = 1, x_3 = 1) = 2/8 = 1/4.$$

The above distribution can be constructed by Hadamard matrices. More generally we show the construction of k-wise independent variables from a small sample space in the appendix. To get a set of k-wise independent bits whose size is n, we need a sample space of size $n^{k/2}$.

In the problem of k-SAT, we only need to make the variables k-wise independent. Thus we need to try $n^{k/2}$ possibilities. The probability of a clause being satisfied maintains $1 - 2^{-k}$. In 3-SAT problem, the number of trials is $O(n^2)$. It efficiently derandomizes the algorithm.

4 Conditional Expectation Method

Another derandomization approach is the conditional expectation method. By the definition of conditional expectations,

$$\begin{aligned} E[\#F] &= Pr(x_1 = 0) \cdot E[\#F|x_1 = 0] + Pr(x_1 = 1) \cdot E[\#F|x_1 = 1] \\ &= \frac{1}{2}E[\#F|x_1 = 0] + \frac{1}{2}E[\#F|x_1 = 1]. \end{aligned}$$

Either $E[\#F|x_1 \leftarrow 0]$ or $E[\#F|x_1 \leftarrow 1]$ must be no less than $E(\#F)$. If $E[\#F|x_1 \leftarrow 1]$ is greater, we set x_1 to true, otherwise set it to false.

Next step we fix x_1 and look at the two conditional expectations by assigning different values to x_2. Following this way we always take the most likely branch at x_i until x_n gets assigned. For $1 \leq i \leq n$, we compute $E[\#F|x_1 \leftarrow b_1, \ldots, x_{i-1} \leftarrow b_{i-1}, x_i \leftarrow 0]$ and $E[\#F|x_1 \leftarrow b_1, \ldots, x_{i-1} \leftarrow b_{i-1}, x_i \leftarrow 1]$ respectively, and take the assignment that makes the expectation larger.

By induction we have

$$E[\#F|x_1 \leftarrow b_1, \ldots, x_n \leftarrow b_n] \geq E[\#F].$$

Because we already know that $E[\#F] \geq \frac{1}{2}m$, our assignment is a $\frac{1}{2}$-approximation.

Algorithm 2 Conditional expectation method

```
1:  function MAXSAT
2:      for i ← 1 to n do
3:          Let E_0 ← E[#F|x_1 ← b_1, ..., x_{i-1} ← b_{i-1}, x_i ← 0]
4:          Let E_1 ← E[#F|x_1 ← b_1, ..., x_{i-1} ← b_{i-1}, x_i ← 1]
5:          if E_0 ≥ E_1 then
6:              b_i ← 0
7:          else
8:              b_i ← 1
9:          end if
10:     end for
11:     return x_1 ← b_1, ..., x_n ← b_n
12: end function
```

Next we show how to compute the conditional expectations. By definition,

$$\begin{aligned} &E[\#F|x_1 \leftarrow b_1, \ldots, x_i \leftarrow b_i] \\ =&\sum_{j=1}^{m} E[Z_j|x_1 \leftarrow b_1, \ldots, x_i \leftarrow b_i] \\ =&\sum_{j=1}^{m} Pr(C_j \text{ is satisfied}|x_1 \leftarrow b_1, \ldots, x_i \leftarrow b_i) \end{aligned}$$

If clause C_j contains a literal that is already assigned to true, then its probability of being satisfied is 1. Otherwise the probability of being satisfied is $1 - (2^{-k'})$, where k' equals the number of unassigned variables in C_j.

Computing each expectation takes time $O(m)$ and we need to compute it $O(n)$ times. The time of this algorithm is $O(mn)$.

Appendix. Construction of pairwise and k-wise independent variables

Construction of pairwise independent variables

We take b independent random bits, which uniformly take values in 0 or 1. Denote the set of random bits by $Y = \{Y_1, \ldots, Y_b\}$. Let $S = \{S_1, \ldots S_{2^b-1}\}$ be all nonempty subsets of $\{1, 2, \ldots b\}$. Let

$$X_{S_j} = \bigoplus_{i \in S_j} Y_i$$

, where $\oplus$ is the exclusive-or operation. We claim that we have constructed pariwise independent bits.

Lemma .1. *The set $\{X_{S_j} | S_j \subseteq S, S_j \neq \emptyset\}$ constructed as above are pairwise independent uniform bits.*

Proof. For two nonempty sets $S_j \neq S_k \subseteq S$,

$$X_{S_j} = X_{S_j \cap S_k} \oplus X_{S_j \setminus S_k}$$
$$X_{S_k} = X_{S_j \cap S_k} \oplus X_{S_k \setminus S_j}$$

$X_{S_j \cap S_k}$, $X_{S_j \setminus S_k}$ and $X_{S_k \setminus S_j}$ are independent variables, because they only depend on disjoint variables in Y. Since S_j and S_k are nonempty, at least two of $S_j \cap S_k$,$S_j \setminus S_k$ and $X_{S_k \setminus S_j}$ are nonempty. Thus the pair X_{S_j} and X_{S_k} take each value in $\{0, 1\}$ with probability $1/4$, indicating they are pairwise independent variables.

□

Other constructions can be found in [2].

Construction of k-wise independent variables

With Vandermonde matrices we can get a set of k-wise independent variables from sample space of size $n^{(k/2)}$[1].

Assume $n = 2^r - 1$. Let $a_1, \ldots, a_{2^r-1}$ be the nonzeroes of $GF(2^r)$. Construct a Vandermonde matrix M_1 as follows.

$$M_1 = \begin{bmatrix} 1 & a_1 & \ldots & a_1^{k-1} \\ 1 & a_2 & \ldots & a_2^{k-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & a_n & \ldots & a_n^{k-1} \end{bmatrix}$$

Take any k rows of M_1 and let the square matrix be V. By properties of Vandermonde matrices, the determinant of the matrix is

$$\det(V) = \prod_{1 \leq i < j \leq n} (a_j - a_i)$$

, which is called Vandermonde polynomial. Because the elements a_i in V are all distinct, the determinant is non-zero. Thus the rank of V is k. So in M_1, any k rows are linearly independent.

Next, construct an $n \times kr$ matrix M_2 by writing each element of $GF(2^r)$ as a polynomial with r terms. Each term has 0 or 1 as its coefficient. We separate each element into a row of r bits in the matrix. Because

in the linear combination of k rows of M_2 reflects the linear combination of corresponding elements in M_1, It is obvious that any k rows of M_2 are still linearly independent.

$$M_2 = \begin{bmatrix} 0\ 0\ \ldots 0\ 1 & (a_1)_1\ \ldots\ (a_1)_r & \ldots & (a_1^{k-1})_1\ \ldots\ (a_1^{k-1})_r \\ 0\ 0\ \ldots 0\ 1 & (a_2)_1\ \ldots\ (a_2)_r & \ldots & (a_2^{k-1})_1\ \ldots\ (a_2^{k-1})_r \\ \vdots & \vdots & \ddots & \vdots \\ 0\ 0\ \ldots 0\ 1 & (a_n)_1\ \ldots\ (a_n)_r & \ldots & (a_n^{k-1})_1\ \ldots\ (a_n^{k-1})_r \end{bmatrix}$$

We wish the sample space to be as small as possible. So next we make the number columns smaller. First delete the first $r-1$ zero columns. Then to save a factor of 2 of the number of columns, delete all even powers in M_2, except for the first column. In this way we finally get a new matrix M.

$$M = \begin{bmatrix} 1 & (a_1)_1\ \ldots\ (a_1)_r & (a_1^3)_1\ \ldots\ (a_1^3)_r & \ldots & (a_1^{k-1})_1\ \ldots\ (a_1^{k-1})_r \\ 1 & (a_2)_1\ \ldots\ (a_2)_r & (a_2^3)_1\ \ldots\ (a_2^3)_r & \ldots & (a_2^{k-1})_1\ \ldots\ (a_2^{k-1})_r \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & (a_n)_1\ \ldots\ (a_n)_r & (a_n^3)_1\ \ldots\ (a_n^3)_r & \ldots & (a_n^{k-1})_1\ \ldots\ (a_n^{k-1})_r \end{bmatrix}$$

Next we show that any k rows of M are linearly indepenent. By contradiction, suppose there exist k rows that are dependent. Because the corresponding rows are independent in M_1, they must be independent on some even power columns, i.e. $\sum_{i\in S} a_i^{2t} \neq 0$, where S is a set of k rows. Since the field has characteristic 2, $\sum_{i\in S} a_i^{2t} = (\sum_{i\in S} a_i^t)^2$. Thus $(\sum_{i\in S} a_i^t)^2 \neq 0$. Continue dividing t in halves until t gets odd. Then there is a set of odd power columns that is linearly independent. In this case, these rows of M must also be independent.

To construct k-wise random bits from matrix M, we randomly take an $(rk/2)$-dimensional vector A. Let $MA = B$. B is an n-dimensional vector where $B_i = \sum_j M_{ij}A_j$.

Because any k rows of M are linearly independent, the corresponding k elements of B are independent. Thus B is a set of k-wise independent bits. To generate B we independently uniformly select all the bits of A. Since there are $rk/2$ bits in A, the sample space is $2^{rk/2} = n^{k/2}$.

It is also known that this is the minimum sample space size. We omit that proof here.

References

[1] N. Alon, L. Babai, and A. Itai. A fast and simple randomized parallel algorithm for the maximal independent set problem. 1985.

[2] M. Mitzenmacher and E. Upfal. *Probability and computing: Randomized algorithms and probabilistic analysis*. Cambridge University Press, 2005.

[3] D. P. Williamson and D. B. Shmoys. *The Design of Approximation Algorithms*. Cambridge University Press, New York, NY, USA, 1st edition, 2011.