ΟΙΚΟΝΟΜΙΚΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΑΘΗΝΩΝ

ATHENS UNIVERSITY
OF ECONOMICS
AND BUSINESS

Special Topics on Algorithms

Fall 2023

Randomized algorithms

Vangelis Markakis – George Zois

markakis@gmail.com

georzois@aueb.gr

Outline

- Introduction to randomized algorithms
- Algorithms for the MAX CUT problem
- Algorithms for MAX-SAT

Randomized Algorithms

- Algorithms that flip coins during their execution
- For example:
  - they may pick a parameter at random from a given range of values (as in primality testing)
  - Or they may take a random yes/no decision during the execution
- They generally do not produce the same output if you run them twice on the same input

Randomized Algorithms

What may be affected by the randomization:

- Running time: Can be polynomial time or polynomial time in expectation or polynomial time with high probability
- Output: Can be wrong (with a small probability)
- Approximability: can be in expectation

Overall a very powerful tool

- For some problems we only know randomized efficient algorithms
- Sometimes deterministic algorithms can be obtained by first designing a randomized algorithm (and then "derandomize" it)

A randomized algorithm for MAX-CUT

The MAX CUT Problem

MAX CUT:

I: An undirected graph G = (V,E) with nonnegative weights on its edges

Q: Find a cut, i.e., a partition (A, B) of V so as to maximize the total weight of the edges that cross the cut

Given a cut (A, B),
w(A, B) = sum of weights of edges crossing the cut

- Applications of MAX CUT: Circuit layout, statistical physics
- Unlike the s-t MIN CUT problem, MAX CUT is NP-complete
- It also does not admit a PTAS

Randomization for MAX CUT

```
Input:= weighted graph G = (V, E) on n vertices
S:= ∅, T:= ∅ //initial partition
 for i=1 to n
        { Flip a coin for vertex $v_i$
          if Head then S:= S ∪ {$v_i$};
          if Tail then T:= T ∪ {$v_i$};
        }
Return the cut (S, T)
```

Analysis of the randomized algorithm

- Let w_{uv} = weight of edge e=(u, v)
- Need first an upper bound on OPT

Claim: OPT $\leq \Sigma_e w_e$

For each edge e, let $X_e = \begin{cases} 1, if\ e\ crosses\ the\ cut \\ 0, \qquad\qquad\qquad otherwise \end{cases}$

- Suppose that the algorithm produced the partition (A, B)
- $w(A, B) = \Sigma_e w_e X_e$
- Hence:

$$E[w(A, B)] = E[\Sigma_e w_e X_e] \underset{\uparrow}{=} \Sigma_e w_e E[X_e] = \Sigma_e w_e \Pr[X_e = 1]$$
$$= ½ \Sigma_e w_e \geq ½\ OPT$$

Linearity of expectation

Analysis of the randomized algorithm

- Can we have a deterministic algorithm with the same approximation ratio?
- YES! Using the method of conditional expectations (more on this later)
  - But it is much easier to obtain first a randomized algorithm for this problem in order to propose and analyze a deterministic algorithm
  - The deterministic algorithm is a "derandomization" of the algorithm we saw
- Is there a better approximation algorithm?
- YES! 0.878-approximation is possible via the use of semidefinite programming (a generalization of linear programming) [Goemans, Williamson 1995]
- Regarding hardness of approximation:
  - Unless P = NP, there is no ratio better than 16/17 [Hastad 2001]
  - Under a stronger but still widely believed conjecture (the unique games conjecture [Khot 2002]), there is no algorithm with ratio better than 0.878 [Khot et al 2007]
  - The same conjecture under which, Vertex Cover also does not admit a better than 2-approximation

Algorithms for MAX-SAT

(CNF) SAT variants

2-SAT: Each clause has 2 literals

Horn SAT: Each clause has at most one positive literal

3-SAT: Each clause has 3 literals

Q: Is there a satisfying truth assignment?

NAE 3-SAT

I: A 3-CNF formula

Q: is there a truth assignment with at least one true and one false literal in each clause ?

1 in 3 3-SAT

I: A 3-CNF formula

Q: is there a truth assignment with exactly one true literal in each clause?

MAX 2-SAT

I: A 2-CNF formula of m clauses and an integer $B \leq m$

Q: is there an assignment satisfying at least B clauses ?

MAX k-SAT

I: A k-CNF formula of m clauses and an integer $B \leq m$

Q: is there an assignment satisfying at least B clauses ?

MAX SAT

The optimization version of SAT problems:

MAX SAT (optimization version)
I: A CNF formula ϕ of m clauses
Q: find a truth assignment satisfying the maximum possible number of clauses

Restrictions of MAX SAT:

MAX k-SAT (optimization version)
I: A k-CNF formula ϕ of m clauses
Q: find a truth assignment satisfying the maximum possible number of clauses

We can also have weights on the clauses and try to maximize total weight

Theorem: Even for k =2, MAX k-SAT is NP-complete

MAX k-SAT

A randomized approximation algorithm for MAX k-SAT:

Algorithm 1: Independently set each variable to: $\begin{cases} 1, & \text{with probality } \frac{1}{2} \\ 0, & \text{with probability } \frac{1}{2} \end{cases}$

- Suppose each clause contains c_i literals
- Suppose also there is a lower bound b on the number of literals, so that $1 \leq b \leq c_i \leq k$
- important: no clause contains a variable and its negation
- Hence, the truth assignment to each literal of a clause is independent of the other literals within the clause

$$\Pr[C_i = 0] = \frac{1}{2} \cdot \frac{1}{2} \cdots \frac{1}{2} = \frac{1}{2^{c_i}}, \qquad \Pr[C_i = 1] = 1 - \frac{1}{2^{c_i}}$$

MAX k-SAT

$X = \#$ of true clauses (random variable)

$$X = X_1 + X_2 + \dots + X_m = \sum_{i=1}^{m} X_i, \qquad X_i = \begin{cases} 1, & \text{if } C_i = 1 \\ 0, & \text{if } C_i = 0 \end{cases}$$

$$E[X_i] = 1 \cdot \Pr[C_i = 1] + 0 \cdot \Pr[C_i = 0] = 1 - \frac{1}{2^{c_i}} \geq 1 - \frac{1}{2^b}$$

$$E[X] = E\left[\sum_{i=1}^{m} X_i\right] = \sum_{i=1}^{m} E[X_i] \geq \sum_{i=1}^{m} (1 - \frac{1}{2^b}) = (1 - \frac{1}{2^b})m$$

$OPT =$ maximum possible # of true clauses, $\quad OPT \leq m$

MAX k-SAT

$$\frac{E[X]}{OPT} \geq \frac{E[X]}{m} = 1 - \frac{1}{2^b} \geq \frac{1}{2}$$ randomized approximate solution

If we know that each clause contains exactly k literals, then we can guarantee a better estimate:

$$\text{If } b = c_i = k, \forall i, \text{ then } \frac{E[X]}{OPT} \geq \frac{E[X]}{m} = 1 - \frac{1}{2^k}$$

Remarks:

- If b ≥ 2, we have a ¾-approximation
- The algorithm’s worst case is when the formula has clauses with single literals

MAX 3-SAT

For 3-CNF formulas:

$$\frac{E[X]}{OPT} \geq 1 - \frac{1}{2^3} = \frac{7}{8} = 0.875$$

Fact 1: for every instance of MAX 3-SAT, the expected # of clauses satisfied by a random assignment is at least 7/8 of the optimal

But, for any random variable, there is some point at which it assumes a value at least as large as its expectation

Thus:

Fact 2 (Implication of the probabilistic technique): for every instance of MAX 3-SAT, there is an assignment satisfying at least 7/8 of all clauses !

MAX 3-SAT

A minor application:

Fact 3: every instance of 3-SAT with at most m ≤ 7 clauses is satisfiable !

Proof:

- By Fact 2, there is an assignment satisfying at least (7/8)m clauses
- So consider such a truth assignment, and let t = number of satisfied clauses
- $t \geq (7/8)\ m$ and $t \leq m$
- For $m < 8$, it holds that $7/8m > m-1$,
- As t is an integer, it follows that t = m!

MAX 3-SAT: Improving Algorithm 1

Fact 2: for every instance of 3-SAT, there is an assignment satisfying at least 7/8 of all clauses !

Algorithm 1 only guarantees this in expectation

Q: Can we find such an assignment? How much running time do we need?

A first attempt:
Algorithm 2: Repeatedly generate random truth assignments until one of them satisfies at least 7m/8 clauses.

How long will it take until we find one by random trials ?

MAX 3-SAT: Improving Algorithm 1

Waiting for the first success (geometric distr.):
Let Z= # of trials until success (a random variable)

/*success probability
p = Pr[a random assignment satisfies at least 7m/8 clauses]

/*Probability mass function
Pr[Z=j] = probability for success in the j-th trial

Pr[Z=j] = $(1-p)^{j-1}p$

$$E[Z] = \sum_{j=1}^{\infty} j \Pr[Z = j] = \sum_{j=1}^{\infty} j(1-p)^{j-1} p = \frac{p}{1-p} \sum_{j=1}^{\infty} j(1-p)^{j}$$

$$= \frac{p}{1-p} \frac{(1-p)}{p^2} = \frac{1}{p}$$

In expectation 1/p random trials suffice

MAX 3-SAT: Improving Algorithm 1

Can we bound 1/p ?

X = # of satisfied clauses by a random assignment; recall E[X] ≥ 7/8m

p_j = Pr[a random assignment satisfies exactly j clauses]

Let m' = the largest integer less than 7/8m, m' < m

$$\sum_{j<7m/8} p_j = 1 - p, \quad \sum_{j\geq 7m/8} p_j = p$$

$$\frac{7}{8}m = E[X] = \sum_{j=1}^{m} jp_j = \sum_{j<7m/8} jp_j + \sum_{j\geq 7m/8} jp_j \leq \sum_{j<7m/8} m' p_j + \sum_{j\geq 7m/8} mp_j$$

$$= m'(1-p) + mp = m' + (m-m')p \leq m' + mp$$

MAX 3-SAT: Improving Algorithm 1

$$\frac{7}{8}m \leq m' + mp \Rightarrow p \geq \frac{7/8m - m'}{m}$$

$$m' = \text{largest integer} < 7/8m \Rightarrow 7/8m - m' \geq 1/8$$

$$\text{Thus, } p \geq \frac{1}{8m} \Rightarrow \frac{1}{p} \leq 8m$$

This implies:

Theorem: there is a randomized algorithm with expected complexity O(m) for finding an assignment satisfying at least 7/8 of all clauses of a 3-SAT instance !

MAX 3-SAT: A second attempt

- Actually it gets even better
- We can “derandomize” Algorithm 1
- Again use the fact that there is always a truth assignment that achieves at least what the expectation says
- Method of conditional expectations:
  - Try to set a variable, say a to 0 or 1
  - One of the two conditional expectations (i.e., given that a = 0 or a = 1) has to be at least as large as the original expectation
    - Hence, we set the variable a accordingly
  - If we know how to compute conditional expectations we are done essentially
  - We repeat until all variables have been set

$$E[X] = E[X \backslash a=0] \cdot Pr[a=0] + E[X \backslash a=1] \cdot Pr[a=1]$$

Hence: there exists a deterministic 7/8-approximation for MAX 3-SAT

Randomized LP rounding for MAX-SAT

MAX SAT: Handling small clauses

- We saw that Algorithm 1 provides better guarantees when the formula does not have “small” clauses
- If we could handle more effectively clauses with a single literal, we can get a better approximation
- We will use Integer Programming for this
- Consider a formula with n variables, $x_1, \ldots, x_n$
- Q: Can we model MAX SAT as an integer program?

Modeling MAX SAT as an integer program

variables $y_i = \begin{cases} 0, FALSE \\ 1, TRUE \end{cases}$ clauses $z_c = \begin{cases} 0, FALSE \\ 1, TRUE \end{cases}$

clause c: $\begin{cases} P_c & \text{literals with positive variables} \\ N_c & \text{literals with negations of variables} \end{cases}$

$if \quad z_c = 1$ then $\begin{cases} \text{at least one variable in } P_c \text{ is 1} \\ \text{OR at least one variable in } N_c \text{ is 0} \end{cases}$

Modeling MAX SAT as an integer program

(IP)

$$\max \sum_{c} z_c$$

$$\sum_{i \in P_c} y_i + \sum_{i \in N_c} (1 - y_i) \geq z_c , \forall c$$

$$z_c \in \{0,1\} \forall c$$

$$y_i \in \{0,1\} \forall i$$

Even if we solve the LP relaxation, how should we do the rounding?

Randomized LP-rounding algorithm for MAX SAT

```
Input:= a CNF SAT formula on variables $x_1,…,x_n$
Solve the LP relaxation
Let $(y_1,…,y_n, z_1,…, z_m)$ be an optimal LP solution
 for i=1 to n
        { Set variable $x_i$ to 1 with
           probability equal to $y_i$
        }
```

Randomized LP-rounding algorithm for MAX SAT

Performance of the LP-based algorithm:

We need to analyze again the quantity:
E[Z] = expected number of satisfied clauses = Σ_c Pr[c is satisfied]

Theorem: For MAX k-SAT, $E[Z] \geq [1 - (1-1/k)^k]$ OPT > (1-1/e) OPT

- Hence 1-1/e = 0.632-approximation algorithm
- This algorithm can also be derandomized via the method of conditional expectations

Final Deterministic algorithm for MAX SAT

```
Input:= a CNF SAT formula on variables $x_1,…,x_n$
1. Run the derandomization of Algorithm 1
2. Run the derandomization of the LP-based
   rounding
3. Return the best of the 2 truth assignments
```

Theorem: The above is a ¾-approximation algorithm for MAX SAT