Probabilistic and Smoothed Analysis of Algorithms
Assignment # 3

May 18, 2023

Roy, Akash | CS22M007
M.Tech CS | Indian Institute of Technology, Madras

Answering any 5.

1. This exercise is to demonstrate the limitations of considering expected running time of an algorithm as a useful measure. For any $n > 0$, describe a function $f : \{0,1\}^n \to \mathbb{N}$ such that

   - $\mathbb{E}[f] = \mathbb{E}_{x\in\{0,1\}^n}[f(x)] = n^c$ for some constant c and
   - $\mathsf{var}[f] = E[f^2] - (E[f])^2 = \Omega(2^n)$.

   I.e., there can be performance measures f which is polynomial in expectation, but variance being exponential. Give formal justification for your answer (i.e., computation of expectation and variance for the function f constructed). Further, write your views on why the function f that you have constructed may/maynot be a good performance measure for any algorithm in practice.

Ans: Suppose f satisfy the following properties

$$f(x) = \begin{cases} O(n^{c'})\ \forall x \in \{0,1\}^n \text{ except for one index} \\ 2^n \text{ at only one } x \end{cases}$$

Then for this function f the expected value is

$$\begin{aligned}\mathbb{E}[f] = \mathbb{E}_{x\in\{0,1\}^n}[f(x)] &= \frac{2^{n-1} * O(n^{c'}) + 2^n}{2^n} \\ &= O(n^c)\end{aligned}$$

for some constant c.

Now if we calculate $\mathbf{var}[f] = \mathbb{E}[f^2] - (\mathbb{E}[f])^2$

$$\begin{aligned}\mathsf{var}[f] &= \mathbb{E}[f^2] - (E[f])^2 \\ &= \frac{n^{2\cdot c'} \cdot 2^{n-1} + 2^{2n}}{2^n} - O(n^c) \\ &= \Omega(2^n)\end{aligned}$$

Suppose f is a performance (time) measures of some algorithm $\mathcal{A}$. Then it says that for all possible inputs over $\{0,1\}^n$ the runtime of the algorithm is bound from above by a polynomial in n.

2. Compute the Fourier expansion of the following Boolean functions. (Note the functions are described in the Boolean/Fourier domain, you need to convert the function to Fourier domain first.)

   (a) Majority function $\mathrm{MAJ}_3\{-1,1\}^n \to \{-1,1\}$, $\mathrm{MAJ}_3(x_1,x_2,x_3) = \mathsf{sgn}(x_1 + x_2 + x_3)$.

Ans: MAJ_3 computes Majority of three boolean variables x_1 and x_2 and x_3.
We can plot the truth table for the MAJ_3 function which is the following,

x_1	x_2	x_3	$\mathrm{Maj}_3(x_1,x_2,x_3)$
1	1	1	1
1	1	-1	1
1	-1	1	1
-1	1	1	1
-1	1	-1	-1
-1	-1	1	-1
1	-1	-1	-1
-1	-1	-1	-1

Following this truth table we can arrive at the equation for Maj_3 via interpolation. For each point (x_1,x_2,x_3) we need to cook a polynomal that is 1 at (x_1,x_2,x_3) and 0 everywhere else.
Such polynomal is the following. As $x_i \in \{-1,1\}$

$$\left(\frac{1+a_1x_1}{2} \cdot \frac{1+a_2x_2}{2} \cdot \frac{1+a_3x_3}{2}\right) = \begin{cases} 1 \text{ if } (a_i) = \mathsf{sgn}(x_i) \text{ and } x_i = |x_i| \\ 0 \text{ otherwise} \end{cases}$$

$$\begin{aligned}\mathrm{Maj}_3(x_1,x_2,x_3) = &(+1)\left(\frac{1+x_1}{2}\cdot\frac{1+x_2}{2}\cdot\frac{1+x_3}{2}\right)+\\ &(+1)\left(\frac{1+x_1}{2}\cdot\frac{1+x_2}{2}\cdot\frac{1-x_3}{2}\right)+\\ &(+1)\left(\frac{1+x_1}{2}\cdot\frac{1-x_2}{2}\cdot\frac{1+x_3}{2}\right)+\\ &(+1)\left(\frac{1-x_1}{2}\cdot\frac{1+x_2}{2}\cdot\frac{1+x_3}{2}\right)+\\ &(-1)\left(\frac{1-x_1}{2}\cdot\frac{1+x_2}{2}\cdot\frac{1-x_3}{2}\right)+\\ &(-1)\left(\frac{1-x_1}{2}\cdot\frac{1-x_2}{2}\cdot\frac{1+x_3}{2}\right)+\\ &(-1)\left(\frac{1+x_1}{2}\cdot\frac{1-x_2}{2}\cdot\frac{1-x_3}{2}\right)+\\ &(-1)\left(\frac{1-x_1}{2}\cdot\frac{1-x_2}{2}\cdot\frac{1-x_3}{2}\right)\end{aligned}$$

Simplifying this will result into the following equation

$$\mathsf{Maj}_3(x_1,x_2,x_3) = \frac{1}{2}x_1 + \frac{1}{2}x_2 + \frac{1}{2}x_3 - \frac{1}{2}x_1\cdot x_2\cdot x_3$$

(b) $NAE_n : \{-1,1\}^n \to \mathbb{R}$, $NAE_n(x) = 1$ if not all of the input bits are equal, and 0 otherwise. Compute the Fourier coefficients for $n = 4$.

Ans: Using interpolation technique we can get the truth table based representation of the function which is the following

$$\begin{aligned}
\mathrm{NAE}_4(x_1,x_2,x_3,x_4) = {}& \frac{1}{16}(1-x_1)(1-x_2)(1-x_3)(1-x_4)\cdot 0\\
&+\frac{1}{16}(1-x_1)(1-x_2)(1+x_3)(1-x_4)\cdot 1\\
&+\frac{1}{16}(1-x_1)(1-x_2)(1+x_3)(1+x_4)\cdot 1\\
&+\frac{1}{16}(1-x_1)(1+x_2)(1-x_3)(1-x_4)\cdot 1\\
&+\frac{1}{16}(1-x_1)(1+x_2)(1-x_3)(1+x_4)\cdot 1\\
&+\frac{1}{16}(1-x_1)(1+x_2)(1+x_3)(1-x_4)\cdot 1\\
&+\frac{1}{16}(1-x_1)(1+x_2)(1+x_3)(1+x_4)\cdot 1\\
&+\frac{1}{16}(1+x_1)(1-x_2)(1-x_3)(1-x_4)\cdot 1\\
&+\frac{1}{16}(1+x_1)(1-x_2)(1-x_3)(1+x_4)\cdot 1\\
&+\frac{1}{16}(1+x_1)(1-x_2)(1+x_3)(1-x_4)\cdot 1\\
&+\frac{1}{16}(1+x_1)(1-x_2)(1+x_3)(1+x_4)\cdot 1\\
&+\frac{1}{16}(1+x_1)(1+x_2)(1-x_3)(1-x_4)\cdot 1\\
&+\frac{1}{16}(1+x_1)(1+x_2)(1-x_3)(1+x_4)\cdot 1\\
&+\frac{1}{16}(1+x_1)(1+x_2)(1+x_3)(1-x_4)\cdot 1\\
&+\frac{1}{16}(1+x_1)(1+x_2)(1+x_3)(1+x_4)\cdot 0
\end{aligned}$$

(c) The inner product function: $IP_{2n} : \{-1,1\}^n \times \{-1,1\}^n \to \{-1,1\}$, given by $IP_{2n}(x,y) = \mathsf{sgn}(\sum_{i=1}^n x_i y_i)$. First write down the Fourier coefficients for the case of $n = 2$ (3 points) and then obtain a general formula (2 points). No proof needed for the general formula.

Ans:

3. Exercise 1.8 in the book Analysis of Boolean Functions by Ryan O'Donell. Point split: 4+2+4.

The (Boolean) dual of $f : \{-1,1\}^n \to \mathbb{R}$ is the function f^t defined by $f^t = -f(-x)$. The function f is said to be odd if it equals its dual, equivalently if $f(-x) = -f(x)\ \forall x$. The function f is said to be even if $f(-x) = f(x)\ \forall x$. Given any function $f : \{-1,1\}^n \to \mathbb{R}$ its odd part is the function $f^{\text{odd}} = \frac{f(x)-f(-x)}{2}$ and its even part is the function $f^{\text{even}} = \frac{f(x)+f(-x)}{2}$

(a) Express $\hat{f}^t(S)$ in terms of $\hat{f}(S)$.

Ans:

(b) Verify that $f = f^{\text{even}} + f^{\text{odd}}$ and that f is odd (respectively, even) if and only if $f = f^{\text{odd}}$.

4. (a) Exercise 1.15 in the book Analysis of Boolean Functions by Ryan O'Donell.

   (b) Let $f : \{-1,1\}^n \to \{-1,1\}$ be a Boolean function. For $k \in [1,n]$, let $f^{=k}$ be the function given by $f^{=k}(x) = \sum_{S\subset[n],|S|=k} \widehat{f}(S)\chi_S(x)$. Prove the formula for $\langle f^{=k}, f^{=\ell}\rangle$ given in Exercise 1.18 of the book " Analysis of Boolean Functions" by Ryan O'Donnel.

Ans: Collaborators: None
We need to show the following

$$\langle f^{=k}, f^{=l}\rangle = \begin{cases} W^k[f] \text{ if } k = l \\ 0 \text{ otherwise} \end{cases}$$

Definition *For $f : \{-1,1\}^n \to \mathbb{R}$ and $0 \le k \le n$ Fourier weight at degree k is the following*

$$W^k[f] = \sum_{\substack{card(S)=k \\ S\subseteq[n]}} \hat{f}(S)^2 \tag{1}$$

Using Definition (1) we can arrive at the inner product asked in the question. From Parseval's theorem we can say that $W^k[f] = ||f^{=k}||_2^2$. Where $f^{=k} = \sum_{card(S)=k} \hat{f}(S)\chi_S$.
Now we calculate $\langle f^{=k}, f^{=l}\rangle$

$$\langle f^{=k}, f^{=l}\rangle = \left\langle \sum_{card(S)=k} \hat{f}(S)\chi_S, \sum_{card(T)=l} \hat{f}(T)\chi_T \right\rangle \tag{2}$$

$$= \sum_{card(S)=k} \sum_{card(T)=l} \hat{f}(S)\hat{f}(T)\langle \chi_S, \chi_T\rangle \tag{3}$$

From class we know that the quantity $\langle \chi_S, \chi_T\rangle = 1$ if $S = T$ hence $\langle \chi_S, \chi_T\rangle = 1$ *iff* $card(S) = card(T) = k$.

Continuing from (3)

$$\langle f^{=k}, f^{=l} \rangle = \sum_{card(S)=k} \hat{f}(S)^2 \tag{4}$$

(4) is defined to be $W^k[f]$. Hence the following is proved.

$$\langle f^{=k}, f^{=l} \rangle = \begin{cases} W^k[f] \text{ if } k = l \\ 0 \text{ otherwise} \end{cases}$$

5. We had stated in the class that Knapsack problem admits a pseudo linear time algorithm. This question is on developing such an algorithm and using it for a simple approximation algorithm for Knapsack.

   (a) Using the standard dynamic programing approach, develop a $O(\mathsf{poly}(n)W)$ time algorithm for the Knapsack problem with n items. Clearly mention each step and give a complexity analysis of the algorithm.

Ans:

Algorithm 1: Knapsack Dynamic Programming Algorithm

```
Procedure knapsack(k,w);
begin
    if $table[k][w] \neq 0$ then
        return $table[k][w]$;
    end
    if $w[k] \leq w$ then
        with = v[k] + knapsack(k-1, w-w[k])
    else
        with = 0
    end
    without = knapsack(k-1,w);
end
$table[k][w] = max\{$with, without$\}$;
return $max$ {with, without}
```

This algorithm runs in $O(n \cdot W)$ time filling each entry in the table at most once where there are nW spaces in the table.

   (b) Using the above algorithm, devlop an algorithm that given ϵ, obtains an $1-\epsilon$ approximate solution(i.e., outputs a solution with profit at least (1-ϵ) times the maximum) in time $\mathsf{poly}(n, 1/epsilon)$.

Ans: Collaborators:

6. Let A be an algorithm for a Euclidean optimization problem $\mathcal{P}$. in the plane. Consider an input instance x and the simple perturbation model with a given parameter $\delta \in (0, 1/2)$. Let $x \in ([0,1]^2)^n$. A δ perturbation of x is given by $(x_1 + y_1, \ldots, x_n + y_n)$, where $y_1, \ldots, y_n$ are chosen uniformly and independent on $[-\delta, \delta]$. The algorithm A is said to be (ϵ, δ) stable with respect to x, if $Pr[|A(x) - A(y)| \leq \epsilon] = 1 - o(1)$, where y is a δ perturbation of x.

   (a) Consider the Christofide's algorithm for ETSP (Euclidean TSP) in the plane. Construct an infinite family of inputs $X = (X_n)_{n>0}$ (i.e., for every $n > 0$, construct an instance X_n with $O(n)$ many points) such that $CHR(X_n) = (1.5 - o(n))TSP(X))$.

   Ans: Collaborators:

   (b) Are the instance you have constructed above (ϵ, δ) stable for suitable values of δ and ϵ? Give a detailed justification to your answer.

   Ans: Collaborators:

7. This question uses the same notion of perturbation as in the above question. Let $\mathcal{P}$ be a Euclidean optimization problem in the plane and $x \in (\mathbb{R})^2)^n$ be an input instance and y be a δ perturbation of x. We say that $\mathcal{P}$ is (ϵ, δ) stable at x, if $Pr[|\mathcal{P}(x) - \mathcal{P}(y)| \leq \epsilon] = 1 - o(1)$.

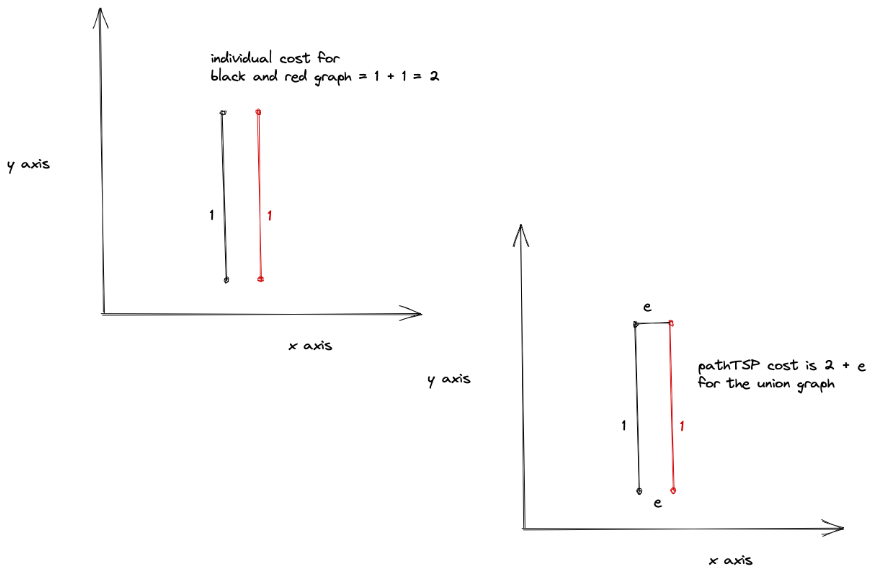
   (a) Suppose $\mathcal{P}$ be an NP hard Euclidean optimization problem such that $\forall x \in (\mathbb{R}^2)^n$, $\mathcal{P}$ is (ϵ, δ) stable at x, for some ϵ and δ. Can we have a smoothed polynomial time algorithm that solves $\mathcal{P}$ exactly? Justify your answer.

   (b) Suppose A is approximation algorithm for $\mathcal{P}$ with some approximation performance γ. Assume that $X = (X_n)$ is an infinite family of worst case instances for A (i.e., approximation ratio of A on X_n is $\gamma - o(1)$). Further, suppose that X_n is (ϵ, δ) stable for $\mathcal{P}$ for every n. What will be the best possible smoothed approximation ratio of A under δ perturbations? Prove your answers.

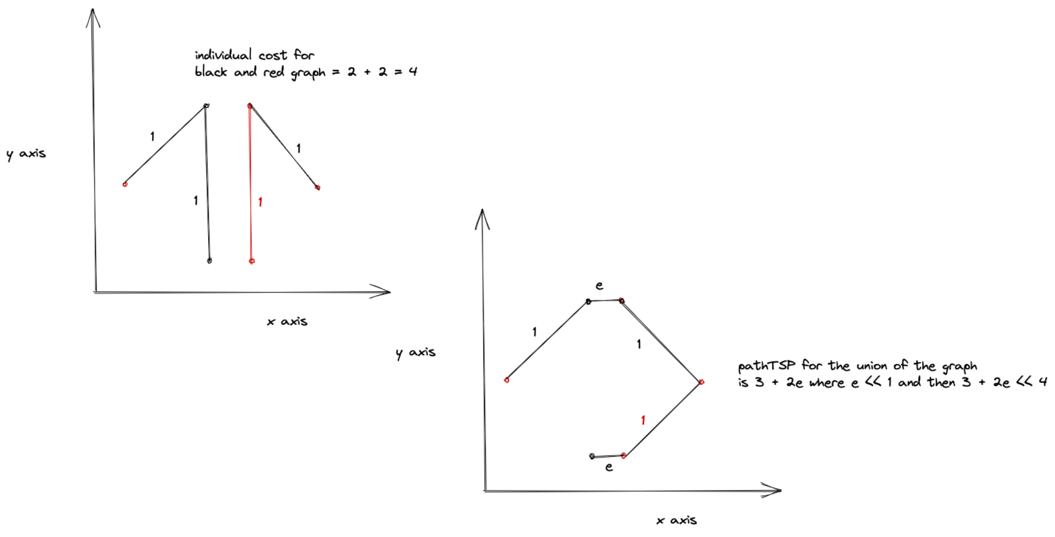
8. Consider the pathTSP problem: Given a set of n points $x_1, \ldots, x_n \in [0,1]^2$, in the Euclidean plane, compute the minimum weight of a Hamiltonian path through the points. Define a Euclidean functional corresponding to pathTSP. Decide if the newly defined function satisfy any of the properties among subadditivity, superadditivity, smoothness, growth bound. Give detailed arguments justifying your answer.

Ans: Collaborators: None

pathTSP shows neither simple superadditivity as well as simple subadditivity. We begin by showing counter examples



Hence union of the graph is a bit higher, and the distance is bounded by $O(diamR)$. Hence this does not show simple subadditivity.


individual cost for
black and red graph = 2 + 2 = 4
y axis
1
1
1
1
x axis
e
y axis
1
1
pathTSP for the union of the graph
is 3 + 2e where e << 1 and then 3 + 2e << 4
1
e
x axis

In the above example the union of the graph's cost is less than the sum of the individual graph's pathTSP cost. Hence this is not showing simple superadditivity.

The functional shows weak form of superadditivity with error terms bounded by $O(diamR)$. This functional also shows monotone properties, hence together we can say this functional shows smoothness.

Bonus

1. Let X be an instance of the k-means clusturing problem, with $|X| = n$. Let $B_1, \ldots, B_r$ be a sequence of clusturings obtained by the k-means algorithm. Show that there is some constant c such that if $r > c$, then at least one of the clusters among the clusterings $B_1, \ldots, B_r$, has seen at least three different configurations. (**Note:** in the class we had shown this for $r > 2^k$, using the pigeonhole principle.)