CS 170 Spring 2017

Efficient Algorithms and Intractable Problems

Prasad Raghavendra and Sanjam Garg

Final Review II-3

1. Cryptography

Let $G = (V, E)$ be a graph that is three-colorable and let c_1, $c_2 : V \to \{R, G, B\}$ be two *valid* coloring functions. c is a valid coloring function if for every $(u, v) \in E$ we have that $c(u) \neq c(v)$. Let $< P, V >$ be the zero-knowledge proof system described in class. Answer True/False for the following questions.

- A malicious verifier learns whether the prover used c_1 or c_2.

  Solution: False. This protocol satisfies zero-knowledge property, which means verifier cannot learn anything from the prover. Therefore it cannot distinguish c_1 and c_2.

- An honest prover in $< P, V >$ can prove to the verifier that the graph is three colorable even without being provided a valid c.

  Solution: False. The prover can only prove it's colorable if the prover knows a valid c.

- An honest verifier in $< P, V >$ accepts a proof only if there exists a three coloring of the considered graph.

  Solution: True. An honest verifier in $< P, V >$ accepts a proof only if the prover knows a three coloring for the graph. Suppose the graph is not three colorable, then it is impossible for an honest verifier to accept a solution.

2. Sampling, Streaming

1. Let $[m]$ denote the set $\{0, 1, \cdots, m-1\}$. For each of the following families of hash functions, say whether or not it is universal, and determine how many random bits are needed to choose a function from the family.

   (a) $H = \{h_{a_1,a_2} : a_1, a_2 \in [m]\}$, where m is a fixed prime and $h_{a_1,a_2}(x_1, x_2) = a_1 x_1 + a_2 x_2 \mod m$. Notice that each of these functions has signature $h_{a_1,a_2} : [m]^2 \to [m]$, that is, it maps a pair of integers in $[m]$ to a single integer in $[m]$.

   Solution: Here H is the same as in the example of pg.46 of the book, only with 2 coefficients instead of 4. With the same reasoning as the proof of the Property in pg.46, we assume that $x_2 \neq y_2$ and we want to determine the probability that equation $a_1(x_1 - y_1) = a_2(y_2 - x_2)$ holds. Assuming we already picked a_1, that probability is equal to $1/m$, since the only way for the equation to hold is to pick a_2 to be $(y_2 - x_2)^{-1} \cdot a_1 \cdot (x_1 - y_1) \mod m$. We can see that, since m is prime, $(y_2 - x_2)^{-1}$ is unique. Thus H is universal. We need $2 \cdot \lceil \log m \rceil$ bits.

   (b) H is as before, except that now $m = 2^k$ is some fixed power of 2.

   Solution: H is not universal, since according to (a) we need a unique inverse of $(y_2 - x_2) \mod m$. For this to hold m has to be prime, which is not true (unless $k = 1$). We need $2k$ bits.

   (c) H is the set of all functions $f : [m] \to [m-1]$.

   Solution: We calculate $P = \Pr[f(x) = f(y)]$ for $x \neq y$. We have $P = \sum_{i=1}^{m-1} \frac{1}{(m-1)^2} = \frac{1}{m-1}$. Thus H is universal. The total number of functions $f : [m] \to [m-1]$ is $(m-1)^m$ so we need $m \log(m-1)$ bits.

2. Given a graph $G = (V, E)$ on n vertices, we would like to estimate the fraction of triples $(i, j, k) \in V$ that form a triangle in G, i.e., (i, j), (j, k) and $(k, i) \in E$. Design a randomized algorithm to estimate this fraction within an error of ± 0.01 with probability 0.99. What is the run-time of your algorithm, in terms of n?

   Solution: We want to use sampling methods. Randomly select t triples independently in the graph, and test how many of those triple satisfy form a triangle, say t'. Then the fraction p is simply t'/t. Based on the lecture notes, we can easily show that this estimate p is unbiased.

   Now, we want to get a error bound for t such that it can satisfy all the error and probability requirement. Recall Chernoff bound:

$$P\left[\left|\frac{1}{t}\sum_{i=1}^{t} X_i - p\right| \geq \varepsilon\right] \leq 2e^{-2\varepsilon^2 t}$$

   where p is the estimated fraction, X_i is a random variable denoting whether triangle inequalities satisfy for each triple we might choose. ε is the error bound. We know from the inequality that the probability of seeing our estimate is outside the ε error bound of the true estimate is less than $2e^{-2\varepsilon^2 t}$.

   In this problem we set $\varepsilon = 0.01$ and probability $2e^{-2\varepsilon^2 t} \leq 1 - 0.99 = 0.01$. Then we solve it for t, which $t \geq 5000 \ln 200$.

   Clearly the runtime is simply $O(t)$, which is independent of n.

3. Given a stream with only one distinct word repeated again and again, can the counting distinct algorithm return a value greater than 1000? If so, what is the probability of this event?

   Solution: If there is only one distinct element, this element will be hashed to a value uniformly between 0 and 1, say k. According to the algorithm, it will output $1/k$ as the final distinct count. The probability of $1/k > 1000$ is equivalent to the probability of $k < 1/1000$. Since k is uniformly distributed between 0 and 1, the probability is simply $1/1000$.

3. Multiplicative Weights

1. In an execution of the weighted majority algorithm with n experts, after t days, the algorithm made ℓ mistakes. What is the largest possible value of the total weight of all experts?

   Solution: First, the number days do not affect the weights. You will only update the weights when you make a mistake. At the beginning the total weights $W = \sum w_i = n$. The weight of the incorrect experts gets reduced by least 1/2 for each incorrect prediction. So the total weight is reduced by a factor of at least 3/4 for every mistake, since the experts who went wrong have more than half the total weight. If the algorithm makes l mistakes the total weight is at most $n(\frac{3}{4})^l$.

2. In an execution of the weighted majority algorithm with n experts, suppose there is an expert (say E_1) who never makes a mistake. What is the maximum number of mistakes the algorithm will make?

   Solution: The worst scenario is for every mistake you make, you are only able to identify the minimum number of bad experts. You can keep reducing their weights until the true expert's weight dominates the rest. The total weights for all experts are n at the beginning. Since the weight for the good expert never changes, essentially, the total changable weight is $n-1$. According to the previous part, it reduces by a factor of at least 3/4 for every mistake. If the algorithm makes l mistakes the total weight is at most $(n-1)(\frac{3}{4})^l$. The true expert's weight never changes and stays at 1. The algorithm will not make any mistakes after the weight for true expert is bigger than the rest. Then you want to have:

$$(n-1)\left(\frac{3}{4}\right)^l < 1$$

Therefore $l > \log_{4/3}(n-1)$. The maximum number of mistakes is $\log_{4/3}(n-1)+1$.

3. "Follow the leader" is a strategy for the experts problem, where on each day, the aggregator always picks the best expert so far, i.e., the expert with the smallest number of mistakes up to day the day.

   Give an example to show that with 3 experts, this strategy could make three times as many mistakes as the best expert.

   Solution: Suppose we have the following three experts, with their predictions on each day. To make it simple, the true result is always 1. If we have more than one experts that have the same smallest number of mistakes, we will choose according to alphabetical order.

Experts	1	2	3	4	5	6	7	8	9
A	0	1	1	0	1	1	0	1	1
B	1	0	1	1	0	1	1	0	1
C	1	1	0	1	1	0	1	1	0

   In this case, we choose experts in the following order: A,B,C,A,B,C,A,B,C. You can see that in this adversarial case, we will never be able to get any correct predictions, since your chosen "best expert" will always make a bad predictions on the day that you choose him/her. In this case, the best experts (all A, B and C) make only 3 mistakes, but you make 9 mistakes, which is three times the mistakes the best expert can make. You can keep growing this table infinitely and generalize it.

4. We are running Probabilistic Multiplicative Weights with $n = 2$ experts for $T = 10,000$ days, and we choose $\varepsilon = 0.01$ Is this close to the optimum choice? (Hint: $\sqrt{\ln 2} \approx .8325$)

   Solution: We need $\sqrt{\frac{\ln n}{T}} \leq \varepsilon$. Plugging in what we know, we have $(\sqrt{\ln 2}/100) \approx .008325$. This is close enough (we also accepted this is not close enough).

5. Consider Probabilistic Multiplicative Weights algorithm, in all of the first 140 days, Expert 1 has cost 0 and Expert 2 has cost 1. In the next day, with what probability will you play Expert 1? Assume $\varepsilon = 0.01$. (Hint: You can assume that $0.99^{70} = \frac{1}{2}$)

   Solution: The weight assigned to expert 1 is $.99^{0\cdot 140} = 1$, while the weight assigned to expert 2 is $.99^{1\cdot 140} \approx 1/4$. So, the probability we play expert 1 is $\frac{1}{1+1/4} = 4/5$.

4. Insider Trading

You are trying to invest in stocks. At the beginning of every day, you buy 1 dollar of stocks, and cash in at the end of the day. We say that a company i has return g_i^t on day t if it earns you a profit of g_i^t dollars per dollar invested.

You have a friend who works at the stock exchange. One day, he hands you a list of n companies (assume n is large), and tells you that for the next T days, one of the companies x (you don't know which one) is going to, on average, earn a profit of 0.5 per day. Additionally, company x will earn a profit of at most 1 per day.

Design a scheme for investing so that at the end of T days, you make at least $0.45T - \frac{\ln n}{0.1}$ dollars. Assume $T \geq 100n$.

Assume for all companies, g_i^t is lower bounded by 0 (you never lose money). Note that the upper bound of 1 dollar only applies to company x.

Solution: This problem can be solved by using experts directly, with a small catch: we only have a guarantee that $g_x^t \leq 1$, but experts requires that all profits be between 0 and 1. To get around this, we simply define profits $h_i^t = \min\{g_i^t, 1\}$.

Now, we simply apply the gain version of experts formula. Define:

$$H \triangleq \sum_{t=1}^{T}\sum_{i=1}^{n} w_i^t h_i^t$$

$$H^* \triangleq \max_j \sum_{t=1}^{T} h_j^t$$

$$G \triangleq \sum_{t=1}^{T} \sum_{i=1}^{n} w_i^t g_i^t$$

$$G^* \triangleq \max_j \sum_{t=1}^{T} g_j^t$$

Using the update rule

$$w_i^{t+1} = w_i^t * (1+\varepsilon)^{h_i^t}$$

we can get a return of

$$H \geq (1-\varepsilon)H^* - \frac{\ln n}{\varepsilon}$$

dollars. Choosing $\varepsilon = 0.1$, and noting that

1. $G \geq H$ (because $\forall_{i,t} g_i^t \geq h_i^t$)
2. $H^* \geq 0.5T$ (because for company x whose profits are always bounded by 1, $h_i^t = g_i^t$. Note that this is why it is important that company x's profits be bounded)

$$\begin{aligned} G &\geq H \\ &\geq 0.9 * H^* - \frac{\ln n}{0.1} \\ &\geq 0.9 * 0.5 * T - \frac{\ln n}{0.1} \\ &= 0.45T - \frac{\ln n}{0.1} \end{aligned}$$