Homework 6
Colorado CSCI 5454

Sai Siddhi Akhilesh Appala

November 30, 2022

People I studied with for this homework: None

Problem 1

Greedy, that tries to add items in order of the most "bang per buck" (value per weight):
Solution- reference from class: **Solution for weighted max cut?**

we need to apply the approximation using the When total randomized algorithm to decide whether which vertex resides in set A or B.

As per the given equation we need to get the maximum of $\Sigma W_{u,v} : u, v \in E, g(u) \neq g(v)$.

- lets say we have 5 vertices namely = $[v_1, v_2, v_3, v_4, v_5]$
- Now we partitioned the vertex based on the weighted max cut, into sets A and B. where $A = v_1, v_2, v_3$ and $B = v_4, v_5$
- we will use the below notation for the problem for the explanation where$g(u) \in A$ defines that vertex U is in set A. if it's B, the notation will change like this $g(u) \in B$.

Totally random Algorithm

- For each vertex, we will check the probability of the vertex getting into set A or set B.
- A we will be checking the above equations whether they reside in the same set or a different set i.e A or B.
- We have 2 sets, A and B, so the probability that a vertex resides in either A or B is $\frac{1}{2}$.
- For the notations please find the below probabilities and their chances of 1 and 0 based on condition.

$$X_{u,v} = 1[g(u) \neq g(v)]$$

$$X_{u,v} = 0[g(u) = g(v)]$$

Let's say ALG defines the randomized algorithm that we are gonna come up with. So we need to find the performance of the algorithm and it's defined as below

$$ALG = \Sigma_{u,v \in E} X_{u,v}$$

we need to do the expected performance of the algorithm which will be indicated as

$$E[ALG] = E[\Sigma_{u,v \in E} X_{u,v}]$$

From the definition of probability, we can say expand the above equation as

$$E[ALG] = E[\Sigma_{u,v \in E} X_{u,v}]$$

$$E[ALG] = \Sigma_{u,v \in E}(0.P_r[X_{u,v} = 0] + 1.P_r[X_{u,v} = 1]])$$

The above equation indicates that the expectation of the algorithm is the sum of 0 multiply the probability of vertex u and b stays in same set and 1 multiply the probability of u, and v stays in a different vertex(which is mentioned in the above equations using$X_{u,v}$

Here we have 4 probabilities of vertex u and v in set A and B, Those are

- $g(u) \in A, g(v) \in A$
- $g(u) \in A, g(v) \in B$
- $g(u) \in B, g(v) \in A$
- $g(u) \in B, g(v) \in B$

$$E[ALG] = \Sigma_{u,v \in E}(0.P_r[X_{u,v} = 0] + 1.P_r[X_{u,v} = 1]])$$

we only need to find the second part of the equation, as 0 multiplies anything is 0.

$$E[ALG] = \Sigma_{u,v \in E}(1.P_r[X_{u,v} = 1]])$$

We need to consider only for the 2nd and 3rd cases of probabilities where they reside in a different set. for a vertex to reside either in A or B is $\frac{1}{2}$.if we consider the 2 cases, we can write like below.

$$E[ALG] = \Sigma_{u,v \in E}(\frac{1}{4} + \frac{1}{4})$$

$$E[ALG] = \Sigma_{u,v \in E}(\frac{1}{2})$$

Which we can write as

$$E[ALG] = \frac{1}{2}[E]$$

To Analyse the approximation of the algorithm, we need to go back and compare it with OPT.

The best OPT can possibly be is the upper bound of it. so we can write it as

$$[E] \geq OPT$$

$$ALG \geq \frac{1}{2}[E]$$

$$ALG \geq \frac{1}{2}OPT$$

Hence we have proved that the approximation of the algorithm is $\frac{1}{2}$ of the optimized one. and approximation factor is $\frac{1}{2}$.

Problem 2

Solution:

We need to showcase the randomized algorithm as an approximation with the optimized one i.e $E[ALG] \geq \alpha OPT$

- For each vertex, we will check the probability of the vertex getting into $S_1, S_2....S_k$ k item sets.
- We have k sets, so the probability that a vertex resides in either $S_1, S_2, S_3...S_k$ is $\frac{1}{k}$.
- For the notations please find the below probabilities and their chances of 1 and 0 based on condition.

$$X_{u,v} = 1[g(u) \neq g(v)]$$

$$X_{u,v} = 0[g(u) = g(v)]$$

From the above explanation, we can say that the probability that vertices U and V stay in the same set is $\frac{1}{k}$

$$P(e_{u,v}[g(u) = g(v)]) = \frac{1}{k}$$

As we know the sum of probability is 1. we can conclude the below statement.

$$P(e_{u,v}[g(u) \neq g(v)]) = 1 - \frac{1}{k}$$

which is equal to below equation

$$\frac{k-1}{k}$$

Assume that ALG specifies the randomized algorithm that we will develop. Therefore, we must determine the algorithm's performance, which is described below.

$$ALG = \Sigma_{u,v \in E} X_{u,v}$$

we need to do the expected performance of the algorithm which will be indicated as

$$E[ALG] = E[\Sigma_{u,v \in E} X_{u,v}]$$

From the definition of probability, we can say expand the above equation as

$$E[ALG] = E[\Sigma_{u,v \in E} X_{u,v}]$$

$$E[ALG] = \Sigma_{u,v \in E}(0.P_r[X_{u,v} = 0] + 1.P_r[X_{u,v} = 1]])$$

$$E[ALG] = \Sigma_{u,v \in E}(0.P_r[X_{u,v} = 0] + 1.P_r[X_{u,v} = 1]])$$

we only need to find the second part of the equation, as 0 multiplies anything is 0.

$$E[ALG] = \Sigma_{u,v \in E}(1.P_r[X_{u,v} = 1]])$$

$$E[ALG] = \sum_{u,v} \frac{k-1}{k} w_{u,v}$$

In the worst-case scenario, we are aware that every edge belongs to a different set. The sum of all edge weights would be the best solution in that scenario. In all other circumstances, the ideal solution would be less than the sum of the edge weights.

To Analyse the approximation of the algorithm, we need to go back and compare it with OPT.

The best OPT can possibly be is the upper bound of it. so we can write it as
From the above equations and explanation, we can conclude that

$$E[ALG] \geq \frac{k-1}{k} OPT$$

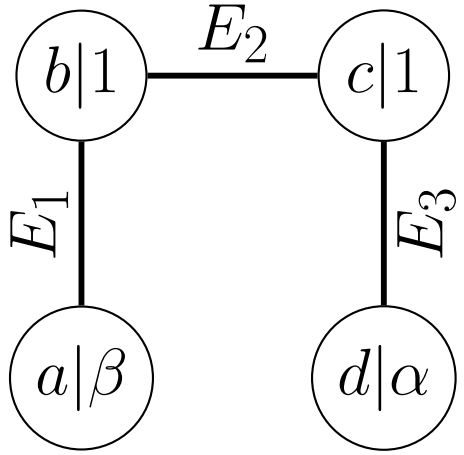
Hence, the algorithm is $\frac{k-1}{k}$ approximated.

Hence we have proved that the approximation of the algorithm is $\frac{k-1}{k}$ of the optimized one. and approximation factor is $\frac{k-1}{k}$.

Problem 3

Part a

Consider following graph as counter example.



We run vertex cover on the above graph

- E_1 arrives and L is empty, so add edge E_1.
- E_2 arrives, but node b is already matched as part of E_1, so ignore.
- E_3 arrives, neither vertex c nor d is included so add edge E_3.

Maximal matching for the above graph is E_1 and E_3. $\therefore$ S = {a, b, d, c}
Vertex weight = $1 + \alpha + 1 + \beta = 2 + \alpha + \beta$

According to the to opt. solution, minimal weight of vertex cover is 1+1 = 2

$$OPT = 2$$
$$\frac{ALG}{OPT} = \frac{\alpha + \beta + 2}{2}$$
$$\frac{ALG}{OPT} = \frac{\alpha + \beta}{2} + 1$$

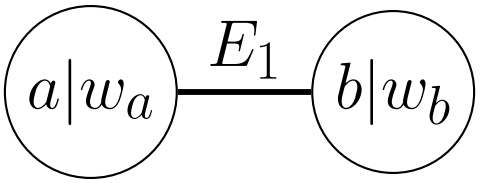
lets say $\frac{\alpha+\beta}{2} = X$ then

$$ALG = (1 + X) * OPT$$

X + 1 is greater than X. SO, our assumption failed. Hence, there exists no approximation factor

Part b

Please find the counterexample below to disprove it
consider the below graph



Weights of vertex a = w_a and vertex b = w_b
let's say w_a and $w_b \geq 1$

The probability of picking a vertex and adding it to the S is $\frac{1}{2}$

$$E[ALG] = E[\Sigma_{u,v \in E} X_{u,v}]$$

$$E[ALG] = \Sigma_{u,v \in E}(w_a.P_r[X_{u,v} = w_a] + w_b.P_r[X_{u,v} = w_b]])$$

The probability of selecting vertex randomly is $\frac{1}{2}$.

$$E[Alg] = \frac{1}{2} * w_a + \frac{1}{2} * w_b$$

$$E[Alg] = \frac{1}{2}(w_a + w_b)$$

OPT will pick the vertex with the lower weight. so if $w_b \geq w_a$
so the ratio will be

$$\frac{ALG}{OPT} = \frac{w_a + w_b}{2.w_a}$$

So it will be w_a for the optimized algorithm. let say $w_a = 1$ and $w_b = 11$ then the equation will be

$$\frac{ALG}{OPT} = \frac{12}{2.1} = 6$$

here it is 6, for the other case, it will be changing. Hence, it is clear that we can't define a proper C for this example.
Hence proved.

Part c

The algorithm discussed in the class is better, as it provides a constant approximation factor of 2, and also considers the probability of vertex weights (i.e for vertex u,v prob of selecting vertex u = $\frac{w_u}{w_u + w_v}$) of every edge, this will help in giving more priority to lower weight. Whereas in the above-randomized algorithm there is no fixed value to c and also we select

the vertex uniformly, which does help to select the optimal value.

$\therefore$ Algorithm discussed in class is better that the randomized algorithm.

$$E[ALG] = \Sigma_{u,v \in E}(w_a.P_r[X_{u,v} = w_a] + w_b.P_r[X_{u,v} = w_b]])$$

$$P_r[X_{u,v}] = \frac{w_a}{w_a + w_b}$$
$$P_r[X_{u,v}] = \frac{w_b}{w_a + w_b}$$

$$E[ALG] = \Sigma_{u,v \in E}(w_a.P_r[X_{u,v} = w_a] + w_b.P_r[X_{u,v} = w_b]])$$

$$E[ALG] = (w_a.\frac{w_a}{w_a + w_b} + w_b.\frac{w_b}{w_a + w_b})$$

$$E[ALG] = \frac{2.w_a.w_b}{w_a + w_b}$$
$$\frac{E[ALG]}{E[OPT]} = \frac{2.w_a.w_b}{(w_a + w_b).w_a}$$

As discussed in the class, and from above derivative, we can conclude that the approximation factor as 2

$$\frac{ALG}{OPT} \leq 2$$

3b-failure explanation The above algorithm failed because it gave both edge vertices an equal chance to be in the vertex cover, even if the edge had the highest weight and the fewest connections. This caused the expected value to increase and prevented the algorithm from achieving a constant approximation factor.

So the algorithm explained in class works better with constant approximation factor than the above one.

Problem 4

Part a

Need to prove: kind of las vegas algorithm:- If we repeat the algorithm n^2 times, the expected number of successes if least 1.

mentioned probabilities Probability of Success(P(S)) for 1 round $\geq \frac{2}{n^2}$

Let ALG number of successes running n^2 times.Now we can conclude below equations

$$E[ALG] = \sum_{i=1}^{n^2} P(S)$$

$$E[ALG] \geq \sum_{i=1}^{n^2} \frac{2}{n^2}$$

$$E[ALG] \geq n^2 * \frac{2}{n^2}$$

$$E[ALG] \geq 2$$

From the above, we can understand that we got at least 1 success if we run algorithm n^2 times.

Part B

As the probability of success is

$$P(s) = \frac{2}{n^2}$$

Then the probability of failure is $P(f)$ will be $1 - p(s)$

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

There is an approximation that is proved when we have an equation with $\forall x \in \mathbb{R}$ (1+x) can be written as e^x

$$(1 + x) \leq e^x$$

When we substitute $-x$ in place of x, even then the below should be true, so the equation will look like

$$1 - x \leq e^{-x}, \forall x \in R \tag{1}$$

From the above approximation, we can modify the current equation as

$$p(f) \leq e^{\frac{-2}{n^2}}$$

As mentioned, if we perform the experiment for n^2 times then the equation will be

$$p(f) = e^{-\frac{2*n^2}{n^2}}$$

We can cancel out the n^2

$$P(F) \leq \frac{1}{e^2}$$

We know that P(S)+p(F) = 1 and as it takes the upper bound we can conclude the below equation

$$P(S) \geq 1 - p(F)$$

$$P(S) \geq 1 - \frac{1}{e^2}$$

Therefore the probability of success, if we repeat the algorithm for n^2 rounds, is at least 1 - $\frac{1}{e^2}$

Part C

As given Probability of success is $1 - \delta$ which means the failure probability is δ. As the sum of the Probability of success and failure is 1.

$$P(s) = \frac{2}{n^2}$$

The probability of failure is $P(f)$ as we know that p(s) + p(f) = 1.

$$p(f) = 1 - p(s)$$

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

As we explained before in the part b $(1 + x) \leq e^x$ and how we derive for $(1 - x) \leq e^{-x}$ And we already proved the below solution for the same.

$$p(f) \leq e^{\frac{-2}{n^2}}$$

If we perform the experiment for X times then we get,

$$p(f) = e^{\frac{-2}{n^2}X}$$

$$P(F) \leq e^{(}\frac{-2}{n^2} * X)$$

We know that P(S)+p(F) = 1

$$p(s) = 1 - p(f)$$

$$P(S) \geq 1 - e^{(}\frac{-2}{n^2} * X)$$

Give that probability of success is $1 - \delta$

$$1 - \delta \geq 1 - e^{(}\frac{-2}{n^2} * X)$$

Remove -1 and multiply with -1 so the equation signs will change

$$\delta \leq e^{(}\frac{-2}{n^2} * X)$$

Apply log to the base e which is ln on both sides.

$$ln(\delta) \leq \frac{-2}{n^2} * X$$

From here we can say that to get the min value we must come up with $X \geq$. so if we change the above equation to

$$X \geq \frac{n^2}{2} ln(\frac{1}{\delta`})$$

From here we can say that we need to run more than or equal to $\frac{n^2}{2} ln(\frac{1}{\delta`})$ times, then we can find the min-cut with a probability of $1 - \delta$.