Sapienza University of Rome

Algorithm Design Second Homework

Andrea Fioraldi 1692419

January 12, 2019

Exercise 1

We redefine our problem as a problem on graphs. Let $G(M, F, E)$ a fully connected graph in which each friend is a node. Each edge (u, v) as a cost $w(u, v) \geq 0$. We define the following functions:
$deg(v, G) := \sum_{u \in G.M \cup G.F} w(v, u)$;
$density(G) := \frac{1}{|C.M \cup G.F|} \sum_{v,u \in G.M \cup G.F} w(v, u)$;

algorithm $michele\_party\_approx(G)$
    $n := |G.M \cup G.F|$
    $H_n := G$
    $max\_d := -1$
    $s := -1$
    for $i = \frac{n}{2}$ **to** 1
        $d := density(H_i)$
        if $d > max\_d$
            $max\_d = d$
            $s = i$
        $m := v \in H_i.M \mid deg(v, H_i) \leq deg(u, H_i)\ \forall u \in H_i.M$
        $f := v \in H_i.F \mid deg(v, H_i) \leq deg(u, H_i)\ \forall u \in H_i.F$
        $H_{i-1} = H_i \setminus \{m, f\}$
    return H_s

Let OPT the optimal solution and $d_{OPT} = \frac{1}{|OPT|} \sum_{v,u \in OPT} w(v, u) = \frac{W}{N}$ its density. Let $m \in OPT \cap M$ and $f \in OPT \cap F$.

Consider that $\frac{W}{N} \geq \frac{W-deg(m)-deg(f)}{N-2} \Leftrightarrow \frac{deg(m)+deg(f)}{N-2} \geq \frac{W}{N-2} - \frac{W}{N} \Leftrightarrow deg(m)+deg(f) \geq W - \frac{N-2}{N} * W = 2 * \frac{W}{N}$. So in OPT the sum of the degree of a generic m and f is at least twice d_{OPT}.

Consider the iteration i of the algorithm in which for the first time two nodes $m \in OPT \cap M$ and $f \in OPT \cap F$ are removed. In the H_i graph all the pairs $m \in H_i \cap M$ and $f \in H_i \cap F$ have $deg(m) + deg(f) \geq 2 * d_{OPT}$ thanks to the fact that we are going to remove nodes (so with minimal sum of degrees) in which the previous observation holds. In H_i there are $\frac{|H_i|}{2}$ pairs that will be removed until $i = 1$. There couples does not share nodes. The total cost of the edges in H_i is greater than $\left(2 * d_{OPT} * \frac{|H_i|}{2}\right)/2$ (the division by 2 is due to the fact that we want to avoid to consider edges twice). So we have that the density is greater than $\frac{2*d_{OPT}*\frac{|H_i|}{2}}{2*|H_i|} = \frac{d_{OPT}}{2}$. Since the algorithm returns the graph with the highest density over all the iterations we have a solution with density at least $\frac{d_{OPT}}{2}$. We proved that *michele_party_approx* is a 2-approximation.
The cost is $O(n^3)$ because the most expensive operation in the body of the loop is the density that cost $|E| = n^2$.

Exercise 2

Our solution was inspired by the Set Cover approximation proof described in [1] but follows the path of the proof shown during the class.
Let A is the set of required skills. S is the set of all the people available, each people is represented as a set of skills $S_j \subseteq A$. Let $n = |A|$.
We can express the Set Cover with Redundancies problem using the following ILP formulation:

$$\begin{aligned} & min \sum_{S_j \in S} c_j * x_j \\ & \text{s.t.} \sum_{S_j | A_i \in S_j} x_j \geq 3 \quad \forall A_i \in A \\ & x_j \in \{0,1\} \qquad\qquad \forall S_j \in S \end{aligned}$$

In order to build a randomized approximation consider the associated LP relaxion where $x_j^* \in [0,1]$. The LP solution is a vector x^* of real values.
Consider the algorithm ALG in which each person S_j is chosen randomly with probability $p_j = min(d * log(n) * x_j^*, 1)$ with all choices that are indepedent. We denote the vector of these choices as x'.
Let $C_i = \sum_{S_j | A_i \in S_j} x'_j$ a random variable that represents the times that the skill A_i is covered. The expectation of C_i is $\mathbb{E}[C_i] = \mathbb{E}\left[\sum_{S_j | A_i \in S_j} x'_j\right] = \sum_{S_j | A_i \in S_j} p_j$. Then, $\mathbb{E}[C_i] = d * log(n) * \sum_{S_j | A_i \in S_j} x_j^* \geq d * log(n) * 3$ because the cover constraint in LP is satisfied.
From this, thanks to the Chernoff lower bound, follows that:

$Pr\Big[C_i < 3\Big] = Pr\Big[C_i < \Big(1-\Big(1-\frac{1}{d*log(n)}\Big)\Big)*d*log(n)*3\Big] \leq Pr\Big[C_i < \Big(1-\Big(1-\frac{1}{d}\Big)\Big)*$
$d*log(n)*3\Big] \leq exp\Bigg(-\frac{1}{2}*3*d*log(n)*\Big(1-\frac{1}{d}\Big)^2\Bigg)$

Note that $\Big(1-\frac{d*log(n)-1}{d*log(n)}\Big)*d*log(n) = 1$. $0 < 1-1/d < 1$ must be valid in order apply Chernoff and so we have that $d > 1$.
We have that the probability that the skill A_i is covered more than 3 times is $Pr[C_i \geq 3] \geq 1-exp\Bigg(-\frac{1}{2}*3*d*log(n)*\Big(1-\frac{1}{d}\Big)^2\Bigg)$.
The probability that at least one A_i is not covered 3 times is $Pr\Big[\bigcup_{A_i\in A} C_i < 3\Big] \leq \sum_{A_i\in A} Pr\Big[C_i < 3\Big] = n*exp\Bigg(-\frac{1}{2}*3*d*log(n)*\Big(1-\frac{1}{d}\Big)^2\Bigg)$.
Note that increasing d give us a better probability that all skills are covered but also, on the other hand, decrease the probability to have a minimal solution because over a threshold we have that all the variables in x' are 1.
The expected cost is $\mathbb{E}\left[\sum_{S_j\in S} c_j * x'_j\right] = \sum_{S_j\in S} c_j * \mathbb{E}[x'_j] = \sum_{S_j\in S} c_j * d*log(n)*x^*_j$ and so it is $d*log(n)$ times the cost of the LP.
With the Markov inequality, we bound with a parameter k the probability to fail on having an average cost of less than $k*d*log(n)$ times the cost of LP.

$$Pr\Big[\sum_{S_j\in S} c_j * x'_j \geq k*d*log(n)*\sum_{S_j\in S} c_j * x^*_j\Big] \leq \frac{\sum_{S_j\in S} d*log(n)*c_j*x^*_j}{\sum_{S_j\in S} k*d*log(n)*c_j*x^*_j} = \frac{1}{k}$$

Obviously this is an event that we want to avoid and setting a big k help us in that but, on the contrary, give us also a worse approximation factor than $d*log(n)$.
We must choose d and k in a way that maximizes the probability that our solution is valid and the expected cost is in the bound and maximize the probability that this solution is minimal.
Choosing $d = 3$ we have an interesting result:
$Pr\Big[\bigcup_{A_i\in A} C_i < 3\Big] = n*exp(-2*log(n)) = 1/n$
The probability that all the skills are covered, so that x' is feasible, is $1-1/n$ that is quite high and so we expect to need only a run to get a feasible solution.
The cost of this approximation must be at most $3*k*log(n)*LP$.
The probability that the solution is not feasible or that the cost exceed the bound is $Pr\Big[\mathbb{E}\left[\sum_{S_j\in S} c_j * x'_j\right] \geq \sum_{S_j\in S} k*d*log(n)*x^*_j \vee \bigcup_{A_i\in A} C_i < 3\Big] \leq \frac{1}{k}+\frac{1}{n}$.
Setting $k = 2$ we get that the solution is feasible and with cost less than $6*log(n)*LP$ with probability greater than $\frac{1}{2}-\frac{1}{2*n}$ that is amost $\frac{1}{2}$ with a reasonable n. We conclude that, as in the normal set cover approximation, the expected number of repetitions are 2 and we choose d and k with the intention to have this a number of repetitions.

Exercise 3

We denote as F^* the optimal solution of the problem. F^* is the minimum cost set of edges that if removed creates k connected components with each target terminal s_i inside each component.
Let $F_i^* \subset F^*$ the set of edges that if removed separated the vertex s_i from the other s_j with $i \neq j$. We have that $\cup_{i=0}^{k} F_i^* = F^*$.
Each edge e in F^* is contained in two F_i^* because it is incident at two connected components. So we can say that:

$$\sum_{i=0}^{k} \sum_{e \in F_i^*} w(e) = 2 * \sum_{e \in F^*} w(e)$$

Our algorithm returns k min cuts F_i. By definition F_i is the minimum set of edges that separates s_i from the other s_j with $i \neq j$ so we have that $\sum\limits_{e \in F_i} w(e) \leq \sum\limits_{e \in F_i^*} w(e)$.

This implies that:

$$\sum_{i=0}^{k} \sum_{e \in F_i} w(e) \leq 2 * \sum_{e \in F^*} w(e)$$

And so, in the worst case, the cost of F is twice the cost of F^*. We proved that this is a 2-approximation algorithm.

Exercise 4

Let k an ordered multiset of genes $g \in G$ such that $g_1 || ... || g_j || ... || g_{|k|} = D$ with $g_j \in k$ ($||$ is the concatenation operator). K is the set of all possible k.
As instance, if $D = ACCA$ we can have $K = \{\{AC, CA\}, \{ACC, A\}\}$ if $G = \{AC, CA, ACC, A\}$.
Let the variables x_i with $i = 1...m$ binary variables that represents if a gene $G_i \in G$ is in D.
Let y_k $\forall k \in K$ a binary variable that tells if the multiset k is used to form D.
The ILP formulation is the following:

$$\begin{array}{lll}
 & min \sum_{i=1}^{m} w_i * x_i & \\
(1) & \sum_{k \in K | G_i \in k} y_k - x_i \leq 0 & \forall i \in \{1...m\} \\
(2) & \sum_{k \in K} y_k \geq 1 & \\
 & x_i \in \{0,1\} & \forall i \in \{1...m\} \\
 & y_k \in \{0,1\} & \forall k \in K
\end{array}$$

(1) means that if a gene G_i is not taken all the k that contains G_i mus be excluded. (2) means that at least one k must be taken, and it is our solution that has minimum cost genes.
Consider now the LP-relaxion with $x_i \geq 0 \wedge y_k \geq 0$. Let's compute the dual of it. In the dual, we have that the number of variables is the same of the number of constraints in the primal $(m+1)$ and vice-versa the number of constraints is the number of variables in the primal $(m+|K|)$. Only constraint (2) contribute to the objective function (cfr. [2]).
When G_i is used the index i is always relative to G and when we use $G_i \in k$ we are considering all the distinct genes in k.

$$\begin{array}{ll} max\ u & \\ \sum_{G_i \in k} v_i - u \geq 0 & \forall k \in K \\ v_i \leq w_i & \forall i \in \{1...m\} \\ u \geq 0, v_i \geq 0 & \forall i \in \{1...m\} \end{array}$$

Exercise 5

The proposed problem looks like a finite zero-sum game theory problem.

Comet, Dasher	Head	Tail
Head	4, -4	-1, 1
Tail	-2, 2	2, -2

We create the Comet's payoff matrix $A_{2,2}$ taking the first value of each table. The matrix $B_{2,2} = -A$, on the other hand, is the payoff matrix for Dasher. Santa must assign the probability to get head or tail for each coin in order to convince both Comet and Dasher to play. To do that the game the game must not be balanced against either player and so we equal the expected payoff of the two players in order to make the game fair.

$$Pr[\text{Comet coins is head}] = p_h \quad Pr[\text{Comet coins is tail}] = p_t \quad p = \begin{pmatrix} p_h \\ p_t \end{pmatrix} = \begin{pmatrix} p_h \\ 1 - p_h \end{pmatrix}$$
$$Pr[\text{Dasher coins is head}] = q_h \quad Pr[\text{Dasher coins is tail}] = q_t \quad q = \begin{pmatrix} q_h \\ q_t \end{pmatrix} = \begin{pmatrix} q_h \\ 1 - q_h \end{pmatrix}$$

The expected payoff of Comet is $p^T * A * q$ and the expected payoff of Dasher is $p^T * B * q = -p^T * A * q$. When they are equal we have that $2 * p^T * A * q = 0$ and so $p^T * A * q = 0$ is out constraint in order to have a fair game.

$\begin{pmatrix} p_h & 1 - p_h \end{pmatrix} * \begin{pmatrix} 4 & -1 \\ -2 & 2 \end{pmatrix} * \begin{pmatrix} q_h \\ 1 - q_h \end{pmatrix} = \begin{pmatrix} p_h & 1 - p_h \end{pmatrix} * \begin{pmatrix} 5 * q_h - 1 \\ -4 * q_h + 2 \end{pmatrix} = p_h * (5 * q_h - 1) + (1 - p_h) * (-4 * q_h + 2) = 9 * p_h * q_h - 3 * p_h - 4 * q_h + 2$

$9 * p_h * q_h - 3 * p_h - 4 * q_h + 2 = 0$ with $p_h, q_h \in [0, 1]$ is the constraint that give to santa a method how to select the probabilities in order to havea fair game.
Note that this equation is a piece of an hyperbola. A valid solution is, as instance, $p_h = 1 \wedge p_t = 0 \wedge q_h = \frac{1}{5} \wedge q_t = \frac{4}{5}$.

Exercise 6

Let h the position of Giorgio's home. Let $S(i) = Pr[Safe|x(t) = i]$ the probability that Giorgio goes to home safely when he is at position i. We know that $Pr[x(t+1) = x(t) + 1] = p$ and $Pr[x(t+1) = x(t) - 1] = 1 - p$. Follows that:

$$S(i) = \begin{cases} 0 & \text{if } i = -1 \\ 1 & \text{if } i = h \\ p * S(i-1) + (1-p) * S(i+1) & \text{otherwise} \end{cases}$$

We have that in general $S(i) = p*S(i-1)+(1-p)*S(i+1)$ and $S(i) = p*S(i)+(1-p)*S(i)$ (from $1 = p - (1-p)$) and so $S(i+1) = \frac{p}{1-p} * \big(S(i) - S(i-1)\big) + S(i)$. Follows that $S(i+2) = \frac{p}{1-p} * \big(S(i+1) - S(i)\big) + S(i+1) = \frac{p}{1-p}\Big(\frac{p}{1-p}\big(S(i) - S(i-1)\big) + S(i) - S(i)\Big) + \frac{p}{1-p} * \big(S(i) - S(i-1)\big) + S(i)$.
With $i = 0$ we have $S(2) = \big(\frac{p}{1-p}\big)^2 * S(0) + \frac{p}{1-p} * S(0) + S(0)$ and so clearly:

$$S(i) = \sum_{j=0}^{i} \Big(\frac{p}{1-p}\Big)^j * S(0).$$

Consider now when $i = h$. The problem says that Giorgio makes an infinite number of steps so we can set $h = +\infty$.

$$1 = S(+\infty) = S(0) * \sum_{j=0}^{+\infty} \Big(\frac{p}{1-p}\Big)^j.$$

This is a geometric series [3]. By definition if $|\frac{p}{1-p}| < 1$ it converges to $\frac{1}{1-\frac{p}{1-p}}$.
The probability that Giorgio goes to the hospital from position 0 is $Pr[Hospital|x(t) = 0] = 1 + Pr[Safe|x(t) = 0] = 1 - S(0)$.
$|\frac{p}{1-p}| < 1 \Leftrightarrow p < 1/2$ and so, finally, we have that:

$$Pr[Hospital|x(t) = 0] = 1 - \frac{1}{\sum_{j=0}^{+\infty} \big(\frac{p}{1-p}\big)^j} = \begin{cases} \frac{p}{1-p} & p < 1/2 \\ 1 & p \geq 1/2 \end{cases}$$

Giorgio goes to hospital for sure when $p \geq \frac{1}{2}$ and goes to hospital with probability at most $\frac{1}{2}$ when $\frac{p}{1-p} \leq \frac{1}{2} \Rightarrow p \leq \frac{1}{3}$.

References

[1] "Randomized rounding - Wikipedia." https://en.wikipedia.org/wiki/Randomized_rounding#Proof. Accessed: 2019-1-3.

[2] "Massimo Roma - Appunti dalla lezione di Ricerca Oprativa, chapter 8." www.dis.uniroma1.it/~roma/didattica/RO17-18/cap8.pdf. Accessed: 2019-1-3.

[3] "Geometric series - Wikipedia." https://en.wikipedia.org/wiki/Geometric_series#Geometric_power_series. Accessed: 2019-1-3.