Homework 11

Name: 方嘉聪 ID: 2200017849

Problem 1.(Max 3-Coloring).
3-Coloring is a yes/no question, but we can phrase it as an optimization problem as follows.

Suppose we are given a graph $G = (V, E)$, and we want to color each node with one of three colors, even if we are not necessarily able to give different colors to every pair of adjacent nodes. Rather, we say that an edge (u, v) is satisfied if the colors assigned to u and v are different.

Consider a 3-coloring that maximizes the number of satisfied edges, and let c^* denote this number. Give a polynomial-time randomized algorithm that produces a 3-coloring that satisfies at least $\frac{2}{3}c^*$ edges in expectation. ◀

Solution. 考虑一个简单的随机算法, 即对 $\forall v \in V$, $\texttt{color}(v)$ 被等概率地赋值为 $1, 2, 3$ 三种颜色. 下面来证明这个算法满足题意:

注意到对于 $\forall e = (u, v) \in E$, $\Pr[\texttt{color}(u) \neq \texttt{color}(v)] = 2/3$, 设随机变量 Y_{uv} 为:

$$Y_{uv} = \begin{cases} 1, & \text{if } \texttt{color}(u) \neq \texttt{color}(v) \\ 0, & \text{Otherwise} \end{cases}$$

则有 $\mathbb{E}[Y_{uv}] = 2/3$. 进而我们有:

$$\mathbb{E}[\text{number of satisfied edges}] = \sum_{(u,v)\in E} \mathbb{E}[Y_{uv}] = \frac{2}{3}|E| \geq \frac{2}{3}c^*.$$

显然算法时间复杂度是多项式的. 证毕. ◁

Problem 2.(MaxSAT).
In class, we designed an approximation algorithm to within a factor of 7/8 for the Max-3SAT problem, where we assumed that each clause has terms associated with 3 different variables. In this problem we will consider the more general Max-SAT problem: given a set of clauses $C_1, \dots, C_k$ over a set of variables $X = \{x_1, \dots, x_n\}$, find a truth assignment satisfying as many of the clauses as possible. Each clause has at least one term in it, but otherwise we do not make any assumptions on the length of the clauses: there may be clauses that have a lot of variables, and others may have just a single variable.

(a) First consider the randomized approximation algorithm we used for Max-3SAT, setting each variable independently to true or false with probability 1/2 each. Show that the expected number of clauses satisfied by this random assignment is at least $k/2$, i.e., half of the clauses is satisfied in expectation. Give an example to show that there are MaxSAT instances such that no assignment satisfies more than half of the clauses.

(b) If we have a clause that consists just of a single term (e.g. a clause consisting just of x_1, or just of $\overline{x_2}$), then there is only a single way to satisfy it: we need to set the corresponding variable in the appropriate way. If we have two clauses such that one consists of just the term x_i, and the other consists of just the negated term $\overline{x_i}$, then this is a pretty direct contradiction.

Assume that our instance has no such pair of "conflicting clauses", that is, for no variable x_i do we have both a clause $C = \{x_i\}$ and a clause $C' = \{\overline{x_i}\}$. Modify the above randomized procedure to improve the approximation factor from 1/2 to at least a 0.6 approximation, that is, change the algorithm so that the expected number of clauses satisfied by the process is at least $0.6k$.

(c) Give a randomized polynomial time algorithm for the general MaxSAT problem, so that the expected number of clauses satisfied by the algorithm is at least a 0.6 fraction of the maximum possible. ◀

Solution. (a) 设一个长度为 s 的子句 $C_i = (x_1, \cdots, x_s)$, $s \geq 1$, 那么有

$$\Pr[C_i \text{ is satisfied}] = 1 - \left(\frac{1}{2}\right)^s \geq 1 - \frac{1}{2} = \frac{1}{2}.$$

设随机变量 Y_i 为:

$$Y_i = \begin{cases} 1, & \text{if } C_i \text{ is satisfied} \\ 0, & \text{Otherwise} \end{cases}$$

则有 $\mathbb{E}[Y_i] \geq 1/2$. 进而我们有:

$$\mathbb{E}[\text{number of satisfied clauses}] = \sum_{i=1}^{k} \mathbb{E}[Y_i] \geq \frac{k}{2}.$$

考虑如下例子 $C = (x_1) \wedge (\overline{x_1})$ 即可.

(b) 当一个公式中不存在矛盾的单子句时, 将上述算法修改为:

- 任意子句的任一变量赋值为真的概率为 p, 赋值为假的概率为 $1-p$.

其中 $p > 1/2$, 我们来证明: 上述算法得到的赋值满足的的子句数的最优赋值的 $\min\{p, 1-p^2\}$ 倍.

对于一个单变量子句 $C_i = \{x_j\}$, 有 (Y_i 定义如上一小问):

$$\Pr[C_i \text{ is satisfied}] \geq p \implies \mathbb{E}[Y_i] \geq p.$$

对于子句 $C_i = (x_1, \cdots, x_s)$, 其中 $s \geq 2$, 不妨设正变量有 a 个, 负变量有 b 个, 那么 $a + b = s$ 且:

$$\Pr[C_i \text{ is satisfied}] = 1 - p^a(1-p)^b \geq 1 - p^s \geq 1 - p^2 \implies \mathbb{E}[Y_i] \geq 1 - p^2.$$

这里利用了 $p > 1/2 > 1 - p$. 那么:

$$\mathbb{E}[\text{number of satisfied clauses}] = \sum_{i=1}^{k} \mathbb{E}[Y_i] \geq \sum_{i=1}^{k} \min\{p, 1-p^2\} = k \cdot \min\{p, 1-p^2\}$$

取 $p = (\sqrt{5}-1)/2$, 有 $p = 1 - p^2 \approx 0.618$, 符合题意, 证毕.

(c) 设形如 (x_i) 的子句个数为 k_1(记为集合 P), 形如 $(\overline{x_i})$ 的子句个数为 k_2(记为集合 N), 不妨设 $k_1 \geq k_2$, 如果 $k_1 < k_2$, 则对于每个 x_i, 取 $\overline{x_i}$ 即可. 注意到对于任意一个赋值, $\forall (x_i) \in P \wedge (\overline{x_i}) \in N$, 至多只能有一个子句可以满足. 设最优解的满足子句数为 k^*, 那么有:

$$k^* \leq k - \min\{k_1, k_2\} = k - k_2$$

那么我们删去 P, N 中所有冲突子句, 在运行 (b) 中的算法. 那么:

$$\mathbb{E}[\text{number of satisfied clauses}] \geq (k - k_2) \cdot \min\{p, 1 - p^2\} \geq k^* \cdot \min\{p, 1 - p^2\} \geq 0.618k^*.$$

证毕.

事实上, 如果使用 LP-relaxation, 可以得到近似比例为 $1 - 1/e \approx 0.632$ 的近似算法. 综合以上两种方法 (等概率选择), 我们可以得到一个近似比例为 3/4 的算法. 这里不赘述了. ◁

Problem 3. (Genome Mapping).
One of the (many) hard problems that arises in genome mapping can be formulated in the following abstract way. We are given a set of n markers $\{\mu_1, \ldots, \mu_n\}$ - these are positions on a chromosome that we are trying to map - and our goal is to output a linear ordering of these markers. The output should be consistent with a set of k constraints, each specified by a triple (μ_i, μ_j, μ_k), requiring that μ_j lie between μ_i and μ_k in the total ordering that we produce (μ_i and μ_k are mutually exchangeable).

Now, it is not always possible to satisfy all constraints simultaneously, so we wish to produce an ordering that satisfies as many as possible. Unfortunately, deciding whether there is an ordering that satisfies at least k' of the k constraints is an NP-complete problem (you don't have to prove this.)

Give a constant $\alpha > 0$ (independent of n) and an algorithm with the following property. If it is possible to satisfy k^* of the constraints, then the algorithm produces an ordering of markers satisfying at least αk^* of the constraints. You should provide a randomized algorithm which has **expected** polynomial running time and **always** produces an approximation within a factor of α. ◀

Solution. 考虑如下算法 (取 $\alpha = 1/3$):

- 随机生成一个 n 个数的排列 π.
- 重复运行上述步骤, 直到输出的排列满足了 αk^* 个约束.

设随机变量 X_i 表示:

$$X_i = \begin{cases} 1, & \text{if 约束} C_i \text{ 被满足} \\ 0, & \text{Otherwise} \end{cases}$$

考虑任意一个约束 $C_i = (\mu_i, \mu_j, \mu_k)$, 那么随机生成的六种排列的概率相同, 其中 2 种排列符合题意, 故 $\mathbb{E}[X_i] = \frac{1}{3}$. 令 $X = \sum_{i=1}^{k} X_i$, 那么有:

$$\mathbb{E}[X] = \sum_{i=1}^{k} \mathbb{E}[X_i] = \frac{k}{3} \geq \frac{k^*}{3}.$$

设 p 为算法输出的排列满足至少 k' 个约束的概率 (即 $p = \sum_{j \geq k'} p_j$), 设 p_j 为满足 j 个约束的概率. 注意到 $k \leq n^3$, 设 $k' = k/3, k'' = \lceil k' \rceil - 1$. 那么有:

$$\begin{aligned} k' = \sum_{j=0}^{k} jp_j &= \sum_{j<k'} jp_j + \sum_{j \geq k'} jp_j \\ &\leq \sum_{j<k'} k''p_j + \sum_{j \geq k'} n^3 p_j \\ &= k''(1-p) + n^3 p. \end{aligned}$$

那么我们有:

$$n^3 p > (n^3 - k'')p \geq k' - k'' \geq \frac{1}{3} \implies p > \frac{1}{3n^3}.$$

上面最后一个不等号是由于, 若 k' 为整数, 那么 $k' - k'' = 1$. 若 k' 不为整数, 讨论 $k' \mod 3$, 即可得到 $k' - k'' \geq 1/3$. 故期望运行时间为 $O(n^3) \cdot \frac{1}{p} \cdot O(n) = \texttt{poly}(n)$, 符合题意, 证毕. ◁

Problem 4. (Dense Induced Graph).
Let $G = (V, E)$ be an undirected graph with n nodes and m edges. For a subset $X \subseteq V$, we use $G[X]$ to denote the subgraph induced on X, that is, the graph $(X, \{(u, v) \in E : u, v \in X\})$.

We are given a natural number $k \leq n$, and are interested in finding a set of k nodes that induces a "dense" subgraph of G; we'll phrase this concretely as follows. Give a polynomial-time algorithm that produces, for a given natural number $k \leq n$, a set $X \subseteq V$ of k nodes with the property that the induced subgraph $G[X]$ has at least $\frac{mk(k-1)}{n(n-1)}$ edges.

You should give a randomized algorithm that has an **expected** polynomial running time and **always** outputs correct answers. ◀

Solution. 考虑随机从 V 中选取 k 个点, 运行时间显然是多项式的. 设随机变量 X_{ij} 为:

$$X_{ij} = \begin{cases} 1, & \text{if } (i,j) \in E \\ 0, & \text{Otherwise} \end{cases}$$

随机选取两个点 $u, v \in V$, 二者有连边的概率为:

$$\Pr[(u,v) \in E] = \frac{m}{\binom{n}{2}} = \frac{2m}{n(n-1)} \implies \mathbb{E}[X_{u,v}] = \frac{2m}{n(n-1)}.$$

那么上述算法的期望输出的边数为:

$$\mathbb{E}[\text{number of edges}] = \sum_{(u,v) \in E} \mathbb{E}[X_{u,v}] = \binom{k}{2} \frac{2m}{n(n-1)} = \frac{mk(k-1)}{n(n-1)}.$$

下面来证明如何得到一个正确解. 考虑如下算法:

- 重复运行上述算法, 直到输出的边数大于等于 $\frac{mk(k-1)}{n(n-1)}$.

设每一次运行得到的边数符合要求的概率为 p, 那么期望的运行时间为 $1/p$, p_j 表示边数恰为 j 的概率. 注意到一个 k 个节点的子图最多有 $e = k(k-1)/2$ 条边, 要求的得到的边数为 $e' = e\frac{2m}{n(n-1)}$, 设边数 $e'' = \lceil e' \rceil - 1$, 那么我们有:

$$\begin{aligned} e' = \sum_{j} jp_j &= \sum_{j<e'} jp_j + \sum_{j\geq e'} jp_j \\ &\leq \sum_{j<e'} e''p_j + \sum_{j\geq e'} ep_j \\ &= e''(1-p) + \binom{k}{2}p. \end{aligned}$$

那么我们有:

$$\left(\binom{k}{2} - e''\right)p \geq e' - e'' \geq \frac{1}{n(n-1)} \implies p \geq \frac{2}{n(n-1)k(k-1)} = \texttt{poly}(n).$$

最后一个不等号是由于, 若 e' 为整数, 那么 $e' - e'' = 1$. 若 e' 不为整数, 注意到 $c = \frac{mk(k-1)}{2} \in \mathbb{Z}$, 而 $e' = c \cdot \frac{2}{n(n-1)}$, 那么有 $e' - e'' \geq \frac{1}{n(n-1)}$.

故综上期望运行时间为 $O(n^3) \cdot \texttt{poly}(n) = \texttt{poly}(n)$, 符合题意, 证毕. $\triangleleft$