高等演算法 HW3

許博翔

May 16, 2024

Notation 1. Let n be a positive integer. $[n] := \{1, 2, \ldots, n\}$.

Problem 0.

Problem 1. Consider $(x_1 \vee x_2) \wedge (x_1 \vee \neg x_2) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_1 \vee \neg x_2)$.

Equivalent ILP:

$\max(z_1 + z_2 + z_3 + z_4)$.

$$\text{subject to} : \begin{cases} y_1 + y_2 \geq z_1 \\ y_1 + 1 - y_2 \geq z_2 \\ 1 - y_1 + y_2 \geq z_3 \\ 1 - y_1 + 1 - y_2 \geq z_4 \\ y_i, z_c \in \{0, 1\} \end{cases}.$$

One can see that in LP, we can set $y_1 = y_2 = \frac{1}{2}$, and get $z_1 = z_2 = z_3 = z_4 = 1$, which maximizes $z_1 + z_2 + z_3 + z_4 = 4$.

But since exactly one of the 4 clauses above must be false, $\max(z_1 + z_2 + z_3 + z_4) = 3$.

$\therefore$ the integrality gap is $\frac{3}{4}$ in this case.

Note that it can't be more than $\frac{3}{4}$ since in the following problem, we'll find a solution ALG that satisfies $\frac{3}{4}OPT \leq \frac{3}{4}OPT(LP) \leq ALG \leq OPT$.

$\therefore$ MAX-SAT has integrality gap $\frac{3}{4}$.

Problem 2.

Lemma 2.1. Let $f(x) = 1 - \frac{1}{4^x} - \frac{3}{4}x$. For $0 \le x \le 1$, $f(x) \ge 0$.

Proof. It's obvious that f is continuous and differentiable in $\mathbb{R}$.
$f'(x) = \ln 4 \frac{1}{4^x} - \frac{3}{4}$.
$\Rightarrow f'(x) > 0 \iff \ln 4 \frac{1}{4^x} > \frac{3}{4} \iff 4^x < \frac{4 \ln 4}{3} \iff x < \log_4(\frac{4 \ln 4}{3}) \approx 0.443$.
$\therefore f$ is increasing in $(-\infty, \log_4(\frac{4 \ln 4}{3}))$ and decreasing in $(\log_4(\frac{4 \ln 4}{3}), \infty)$.
$\Rightarrow \forall x \in [0, \log_4(\frac{4 \ln 4}{3})], x \ge f(0) = 0$, and $\forall x \in [\log_4(\frac{4 \ln 4}{3}), 1], x \ge f(1) = 0$.
$\therefore f(x) \ge 0$ for all $x \in [0, 1]$. ■

Let c be a clause.
The probability that c is satisfied $= 1 - \prod_{i \in S_c^+} (1 - 4^{y_i^* - 1}) \prod_{i \in S_c^-} (1 - (1 - 4^{y_i^* - 1})) \overset{\because 1 - 4^{y_i^* - 1} \le 4^{-y_i^*}}{\ge}$
$1 - \prod_{i \in S_c^+} 4^{-y_i^*} \prod_{i \in S_c^-} 4^{y_i^* - 1} = 1 - (\frac{1}{4})^{\sum_{i \in S_c^+} y_i^* + \sum_{i \in S_c^-} (1 - y_i^*)}$.
By the restrictions in LP, there is $\sum_{i \in S_c^+} y_i^* + \sum_{i \in S_c^-} (1 - y_i^*) \ge z_c^*$.
$\therefore$ the probability that c is satisfied $\ge 1 - (\frac{1}{4})^{\sum_{i \in S_c^+} y_i^* + \sum_{i \in S_c^-} (1 - y_i^*)} \ge 1 - (\frac{1}{4})^{z_c^*} \overset{\text{by } \textbf{Lemma (2.1)}}{\ge}$
$\frac{3}{4} z_c^*$.
$\therefore$ the expected number of clauses that are satisfied $\ge \frac{3}{4} \sum_c z_c^*$.

Problem 3. Let V denote the vertex set, and E denote the edge set.

Algorithm:

For every vertex, color it with one of the k colors uniform randomly and independently.

For every edge (u, v), $\Pr[(u, v) \in S] = \Pr[u, v \text{ have the different colors }] = 1 - \frac{1}{k}$.
$\therefore$ the expected size of S is $(1 - \frac{1}{k})|E| \ge (1 - \frac{1}{k})OPT$, and this is a randomized $(1 - \frac{1}{k})$-approximation algorithm.

Derandomize:

Suppose that $V = [n]$.

Let $[k]$ denote the k colors.

Run the following algorithm with parameter m to obtain the coloring $c_m : [n] \to [k]$.

When $m = n$, the algorithm is deterministic.

- for $i = 1$ to m:
  - Choose j s.t. $|\{1 \le i' \le i-1 : c_m(i') \ne j, (i', i) \in E\}|$ is maximized. – (1)
  - Set $c_m(i) = j$ (i.e. color the vertex i with j).
- for $i = m+1$ to n:
  - Uniformly choose j from k.
  - Set $c_m(i) = j$ (i.e. color the vertex i with j).

Let the resulting S of the algorithm be S_m.
$\mathrm{E}[|S_0|] \ge (1 - \frac{1}{k})OPT$, which has been proved above.
$\mathrm{E}[|S_i|] = |\{(u,v) \in E : u < v < i, c_i(u) \ne c_i(v)\}| + |\{(u,i) \in E : u < i, c_i(u) \ne c_i(i)\}| + (1 - \frac{1}{k})|\{(u,v) \in E : u < v, v > i\}| \overset{\text{By (1)}}{\ge} |\{(u,v) \in E : u < v < i, c_i(u) \ne c_i(v)\}| + (1 - \frac{1}{k})|\{(u,i) \in E : u < i\}| + (1 - \frac{1}{k})|\{(u,v) \in E : u < v, v > i\}| = \mathrm{E}[|S_{i-1}|]$.
$\therefore$ the algorithm with parameter $m = n$, which is a deterministic algorithm, satisfied $|S_n| \ge |S_{n-1}| \ge \cdots \ge |S_0| \ge (1 - \frac{1}{k})OPT$.
Clearly, setting $c_n(i) = j$ above is $O(1)$.
One can first store the neighborhood of each vertex, and then in (1), run though all neighbors of i.
Since each edge will be run for twice in (1), the running time of this algorithm is $O(|V| + |E|)$.

Problem 4. Let $\mathrm{OPT} = \sum_i f_i y_i = \sum_j \alpha_j$. (They're equal by the strong duality theorem).
$\Rightarrow$ all slackness conditions must hold.
$\Rightarrow \alpha_j - \beta_{ij} = c_{ij}$ or $x_{ij} = 0$, $\forall i, j$.
Let p_j denote the falicity that serves j, q_i denote the chosen client j that open i in N_j, and $B_i := \{j : p_j = i\}$.
If $p_j \notin N_j$, it means that $N_j \cap N_{q_{p_j}} \ne \emptyset$. Let r_j denote a facility that $\in N_j \cap N_{q_{p_j}}$.
Else just simply set $r_j := p_j$.
Let F denote the set of falicities that are open.

$\sum_{i\in F} f_i \le \sum_{i\in F} f_i \sum_{k\in N_{q_i}} x_{kq_i} \le \sum_{i\in F} f_i \sum_{k\in N_{q_i}} y_k \overset{f_i\le f_k,\ \forall k\in N_{q_i}}{\le} \sum_{i\in F}\sum_{k\in N_{q_i}} f_k y_k \overset{N_{q_i}\cap N_{q_k}=\emptyset,\ \forall i\ne k}{\le}$
$\sum_i f_i y_i = OPT.$

$\forall j,\ c_{p_j j} \le c_{r_j j} + c_{r_j q_{p_j}} + c_{p_j q_{p_j}}$ (by the definition of metric).

Since q_{p_j} is chosen before j, there is $\alpha_{q_{p_j}} \le \alpha_j$.

Since $r_j \in N_j, r_j \in N_{q_{p_j}}, p_j \in N_{q_{p_j}}$, by the definition of N, there is $x_{r_j j}, x_{r_j q_{p_j}}, x_{p_j q_{p_j}}$ are all nonzero.

$\because$

$c_{r_j j} \le c_{r_j j} + \beta_{r_j j} = \alpha_j.$

$c_{r_j q_{p_j}} \le c_{r_j q_{p_j}} + \beta_{r_j q_{p_j}} = \alpha_{q_{p_j}} \le \alpha_j.$

$c_{p_j q_{p_j}} \le c_{p_j q_{p_j}} + \beta_{p_j q_{p_j}} = \alpha_{q_{p_j}} \le \alpha_j.$

$\Rightarrow c_{r_j j} \le 3\alpha_j.$

$\therefore \sum_{i\in F}\sum_{j\in B_i} c_{p_j j} \le \sum_{i\in F}\sum_{j\in B_i} 3\alpha_j = 3\sum_j \alpha_j = 3OPT.$

$\therefore \sum_{i\in F} f_i + \sum_{j\in B_i} c_{p_j j} \le OPT + 3OPT = 4OPT.$

Since OPT is the optimal solution of LP relaxation, which is not greater than the optimal solution of ILP (denote as OPT').

$\therefore \sum_{i\in F} f_i + \sum_{j\in B_i} c_{p_j j} \le 4OPT \le 4OPT'.$

$\Rightarrow$ this is a 4-approximation.

Problem 5. We'll use the term "at time t" denote when the value of α_j of unserved client j is set to t in the algorithm (that is, not performing 1. or 2. yet).

Let $U^{(t)}$ denote U at time t.

Let p_j denote the facility that serves j, and $B_i := \{j : p_j = i\}$.

Suppose that f_i is open at time a_i.

1. (a) while there are unserved clients
   - i. for i in facilities
     - A. if i is closed, find a set of unserved clients $S(i)$ s.t. $val(i) := \dfrac{f_i + \sum_{j\in S(i)} c_{ij}}{|S(i)|}$ is minimized. (This is equivalent to 1. in the algorithm.)
     - B. if i is open, find an unserved client $S(i) = \{s(i)\}$ s.t. $val(i) :=$

$c_{is(i)}$ is minimized. (This is equivalent to 2. in the algorithm.)

ii. Let i^* be a facility s.t. $val(i^*)$ is minimized.

iii. Open i^* if it's closed, and serve all clients in $S(i^*)$ by i^*.

2. Lemma: if $\alpha_j > c_{ij}$, then $\alpha_j \leq a_i$.

   Proof: If $\alpha_j > a_i$, then j is served after i is open. By 2. in the algorithm, $\alpha_j \leq c_{ij}$.

   $\therefore$ the lemma holds.

   There are 2 cases:

   Case 1: $\alpha_k = 0$.

   In this case, $\alpha_j = c_{ij} = 0, \ \forall j \in \{1, 2, \ldots, k\}$.

   $\therefore \displaystyle\sum_{j=x}^{k}(\alpha_x - c_{ij}) = 0 \leq f_i$ holds for all $x = 1, 2, \ldots, k$.

   Case 2: $\alpha_k \neq 0$.

   $\Rightarrow \alpha_k > c_{ik}$.

   $\Rightarrow$ by the lemma, $a_i \geq \alpha_k$.

   At time α_x, $x, x+1, x+2, \ldots, k$ are unserved, by 1. in the algorithm, $\displaystyle\sum_{j=x}^{k}(\alpha_x - c_{ij}) \leq \sum_{j \in U^{(\alpha_x)}} \max(0, \alpha_x - c_{ij}) \leq f_i$. $\therefore \displaystyle\sum_{j=x}^{k}(\alpha_x - c_{ij}) \leq f_i$ always holds.

3. Claim: $\alpha_j - \alpha_x \leq c_{ix} + c_{ij}, \ \forall 1 \leq x \leq j \leq k$.

   Proof: If $\alpha_j = \alpha_x$, then the claim holds trivially.

   If $\alpha_j > \alpha_x$, then $\alpha_j > \alpha_x \geq a_{p_x}$ since p_x is open before x is served.

   $\Rightarrow$ by the lemma, $\alpha_j \leq c_{p_x j}$.

   $\Rightarrow \alpha_j - \alpha_x \leq c_{p_x j} - \alpha_x \overset{\text{metric}}{\leq} c_{ij} + c_{ix} + c_{p_x x} - \alpha_x \overset{x \text{ is served by } p_x}{=} c_{ij} + c_{ix}$.

   $\Rightarrow \displaystyle\sum_{j=x}^{k}(\alpha_j - c_{ix} - 2c_{ij}) \overset{\text{the claim}}{\leq} \sum_{j=x}^{k}(c_{ix} + c_{ij} + \alpha_x - c_{ix} - 2c_{ij}) = \sum_{j=x}^{k}(\alpha_x - c_{ij}) \leq f_i$.

4. $\displaystyle\sum_{j=1}^{k}(\alpha_1 - c_{ij}) \leq f_i$.

   $\displaystyle\sum_{j=1}^{k}(\alpha_j - c_{i1} - 2c_{ij}) \leq f_i$.

   Since $\alpha_1 - c_{i1} \geq \alpha_1 - 3c_{i1} \geq 0$.

$$\therefore \sum_{j=1}^{k}(\alpha_j - 3c_{ij}) \leq \sum_{j=1}^{k}(\alpha_1 - c_{ij} + \alpha_j - c_{i1} - 2c_{ij}) \leq 2f_i.$$

5. We need to define $\alpha'_j, \beta'_{ij} := \max(\alpha'_j - c_{ij}, 0)$ so that $\sum_j \beta'_{ij} \leq f_i$ can be satisfied for all i.
   Let $\alpha'_j = \frac{1}{3}\alpha_j$, one can see that $\sum_j \beta'_{ij} = \sum_{j:\alpha_j \geq 3c_{ij}} \alpha'_j - c_{ij} = \sum_{j:\alpha_j \geq 3c_{ij}} \frac{1}{3}(\alpha_j - 3c_{ij}) \leq \frac{2}{3}f_i \leq f_i$.
   From 1. of the algorithm, $\sum_{j \in B_i}(\alpha_j - c_{ij}) = f_i$.
   $\Rightarrow \sum_{j \in B_i} c_{ij} + f_i = \sum_{j \in B_i} \alpha_j \leq 3 \sum_{j \in B_i} \alpha'_j,\ \forall i.$
   $\therefore$ this is a 3-approximation.