Midtern Exam

Wenting Li liw14@rpi.edu

October 7, 2017

(1)

Let define variables $x_i, i = 1, \dots, n$ for the ith spy, and $x_i = 1$ if the ith spy is chosen in the team and $x_i = 0$ if he is not in the team; define the variables $y_j, j = 1, \dots, m$ for the j task, and $y_j = 1$ if the jth task is assigned to S_j to do and $y_j = 0$ if the jth task is assigned to T_j, then we can model this problem by an integer programming issue:

$$\begin{aligned}
\text{minimize } & \Sigma_{i=1}^n x_i && (1)\\
\text{subject to } & x_i \geq y_j \quad \text{if } i \in S_j && (2)\\
& x_i \geq 1 - y_j \quad \text{if } i \in T_j && (3)\\
& x_i \in \{0, 1\} && (4)\\
& y_j \in \{0, 1\} \quad i = 1, \dots, n, j = 1, \dots, m && (5)
\end{aligned}$$

I first explain the equivalence of this Integer programming issue and the issue of selecting spies to finish the tasks. As our objective function is to minimize the sum of x_i, then from (3) and (4) we know that x_i is exactly 1 if the jth task is assigned to S_j and $i \in S_j$, or if the jth task is assigned to T_j and $i \in T_j$. Thus no mater the jth task is assigned to S_j or T_j, all the members i in that set are selected in the team, hence the solution is feasible to finish all the tasks. Therefore, the minimum of the objective function returns the minimum number of the spies in the team.

Then we can obtain the linear problem relaxation by replacing $x_i, y_j \in \{0, 1\}$ with $0 \leq x_i \leq 1, 0 \leq x_i \leq 1$, then we have

$$\begin{aligned}
\text{minimize } & \Sigma_{i=1}^n x_i && (6)\\
\text{subject to } & x_i \geq y_j \quad \text{if } i \in S_j && (7)\\
& x_i \geq 1 - y_j \quad \text{if } i \in T_j && (8)\\
& 0 \leq x_i \leq 1 && (9)\\
& 0 \leq y_j \leq 1 \quad i = 1, \dots, n, j = 1, \dots, m && (10)
\end{aligned}$$

By solving this linear programming problem, we can obtain its optimal solution x_i^* and y_j^* and LP-OPT.

Thus my approximation algorithm is to choose the set S_j with the probability y_j^* and choose the set T_j with the probability $1 - y_j^*$ for each task j until all the tasks are assigned either by S_j or T_j.

This algorithm can be achieved in polynomial time, and we can demonstrate that this is a 2-approximation algorithm.

Proof:

The expectation of the size of our team is:

$$\begin{aligned}
E(\Sigma_{i=1}^n x_i) &= \Sigma_{i=1}^n E(x_i) && (11)\\
&= \Sigma_{i=1}^n \text{Pr}(x_i \text{ is chosen in the team}) && (12)\\
&= \Sigma_{i=1}^n (1 - \text{Pr}(x_i \text{ is not chosen in the team})) && (13)
\end{aligned}$$

Assume that there are k_i sets (in all the sets S_j and $T_j, j = 1, \ldots, m$) that contain the spy x_i

$$\text{Pr}(x_i \text{ is not chosen in the team}) = \prod_j \text{Pr}(S_j \text{ is not chosen when } i \in S_j) \prod_j \text{Pr}(T_j \text{ is not chosen when } i \in T_j) \tag{14}$$

$$= \prod_{j:i \in S_j} (1 - y_j^*) \prod_{j:i \in T_j} y_j^* \text{ Since (7) and (8), then} \tag{15}$$

$$\geq \prod_{j:i \in S_j} (1 - x_i^*) \prod_{j:i \in T_j} (1 - x_i^*) \tag{16}$$

$$= (1 - x_i^*)^{k_i} \tag{17}$$

where $k_i \geq 1, 0 \leq x_i^* \leq 1$,

Then (13) satisfy:

$$E(\Sigma_{i=1}^n x_i) = \Sigma_{i=1}^n (1 - \text{Pr}(x_i \text{ is not chosen in the team})) \tag{18}$$

$$\leq \Sigma_{i=1}^n (1 - (1 - x_i^*)^{k_i}) \tag{19}$$

$$\leq \Sigma_{i=1}^n 2x_i^* \tag{20}$$

$$= 2 \text{ LP-OPT} \tag{21}$$

Notice that the demonstration of (20) is following:

Let function $f(x) = 1 - (1 - x)^k - 2x$, where $x \in [0, 1], k \geq 1$, then the derivative of $f'(x) = k(1 - x)^{k-1} - 2 \leq k - 2$, thus there are two conditions:

1. If $k \leq 2$, then $f'(x) = k(1 - x)^{k-1} - 2 \leq k - 2 \leq 0, \Rightarrow f(x)$ is a decreasing function and $f(x) \leq f(0) = 0$, thus $1 - (1 - x)^k - 2x \leq 0 \Rightarrow 1 - (1 - x_i^*)^{k_i} \leq 2x_i^*$ since $x_i^* \in [0, 1], k_i \geq 1$

2. If $k > 2$, then we have

$$f(x) = 1 - (1 - x)^k - 2x \tag{22}$$

$$< 1 - (1 - x)^2 - 2x \tag{23}$$

$$= -x^2 \tag{24}$$

$$\leq 0 \tag{25}$$

Therefore, we can conclude that $f(x) = 1 - (1 - x)^k - 2x \leq 0, \Rightarrow 1 - (1 - x)^k \leq 2x \Rightarrow \Sigma_{i=1}^n (1 - (1 - x_i^*)^{k_i}) \leq \Sigma_{i=1}^n 2x_i^*$

(2)

(a)

Let $N(v) = \{w \in V | (v, w) \in E\} \cup \{v\}$ and x_v for each $v \in V$, then define the set $\mathcal{N} = \{N(v) | \forall v \in V, N(v) = \{w \in V | (v, w) inE\} \cup \{v\}\}$, we can define a LP-relaxation and its dual for the DOMINANTING SET problem as follows:

$$\text{minimize } \Sigma_{v \in V} x_v \omega(v) \tag{26}$$

$$\text{subject to } \Sigma_{v: v \in N(v)} x_v \geq 1 \quad \forall N(v) \in \mathcal{N} \tag{27}$$

$$x_v \geq 0, \quad \forall (v) \in V \tag{28}$$

and its dual problem is:

$$\text{maximize } \Sigma_{N\in\mathcal{N}} y_N \tag{29}$$
$$\text{subject to } \Sigma_{N:v\in N(v)} y_N \le \omega(v), \quad \forall v \in V \tag{30}$$
$$y_N \ge 0, \quad \forall N(v) \in \mathcal{N} \tag{31}$$

(b)

Algorithm 1 The primal-dual method of DOMINATING SET

Initialization: $y \leftarrow 0, C \leftarrow \emptyset$
while there are some $N(v)$ that has no vertex in the set C **do**
  Pick a set $N(v) \in \mathcal{N}$ has no vertex in C
  For any one vertex $u \in N(v)$, increase the y_N of all the sets $N \in \mathcal{N}$ that contains u $u \in N$
  until for one node $\bar{u} \in N(v)$ satisfies that $\Sigma_{N:\bar{u}\in N} y_N = \omega(\bar{u})$
  $C \leftarrow C \cup \{(\bar{u})\}$
return C ▷ The minimum-weight doimnant set C

Claim 1: This is a Δ-approximation algorithm.

Proof:
The total weight of our algorithm satisfy:

$$\begin{aligned}
\Sigma_{v\in C}\omega(v) &= \Sigma_{v\in C}\Sigma_{N:v\in N} y_N & (32)\\
&\le \Sigma_{N\in\mathcal{N}} |N| y_N & (33)\\
&= \Sigma_{N\in\mathcal{N}:v\in N} |\text{degree} + 1| y_N \text{ As the maximum degree is } \Delta & (34)\\
&\le \Sigma_{N\in\mathcal{N}} |\Delta + 1| y_N & (35)\\
&\le |\Delta + 1|\text{OPT} & (36)
\end{aligned}$$

Notice that in each iteration of our method, we only add a single new node $\bar{u} \in N(v)$ to C , and $N(v)$ is the set that has no vertex in C. As the degree of any one set is ≥ 1, there are at least two vertexes in each $N(v)$. Then after we add one vertex $\bar{u}$ of $N(v)$ to set C, there remain at least one vertex not in C, but we will never add any vertex of set $N(v)$ to C since $N(v)$ already has one vertex in C, therefore there are at most Δ vertexes of each $N(v)$ added to set C, then we have:

$$\begin{aligned}
\Sigma_{v\in C}\omega(v) &= \Sigma_{v\in C}\Sigma_{N:v\in N} y_N & (37)\\
&= \Sigma_{N\in\mathcal{N}}\{\text{The number of vertexes of } N \text{ are added to set } C\}\ y_N & (38)\\
&\le \Sigma_{N\in\mathcal{N}} |\Delta| y_N & (39)\\
&\le |\Delta|\text{OPT} & (40)
\end{aligned}$$

Hence, this is a Δ-approximation algorithm.