CSE 632: Analysis of Algorithms II **Fall 2017**

Lecture 7 (09/20/2017): Linear Programming

Lecturer: Shi Li *Scribe: Yuze Liu*

7.1 Linear Programming Basic

First we have a simple example for Linear Programming:

$$\min\ (3x_1 + 2x_2) \text{ s.t}$$

$$\begin{cases} x_1 + x_2 \geq 3 \\ 2x_1 - 3x_2 \geq 5 \\ x_1 + 4x_2 \geq 6 \\ x_1, x_2 \geq 0 \end{cases}$$

A more general form of the linear programming is:

$$\min\ (c_1x_1 + c_2x_2 + \cdots + c_nx_n) \text{ s.t.}$$

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \geq b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \geq b_2 \\ \vdots \\ a_m x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \geq b_m \\ x_1, x_2, x_3, \cdots, x_n \geq 0 \end{cases}$$

And we can write the above expression in a more simple way:

$$c = \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{pmatrix} \qquad x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \qquad A = \begin{pmatrix} a_{11}, a_{12}, a_{13}, \cdots, a_{1n} \\ a_{21}, a_{22}, a_{23}, \cdots, a_{2n} \\ \vdots \\ a_{m1}, a_{m2}, a_{m3}, \cdots, a_{mn} \end{pmatrix} \qquad b = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}$$

$$\min(c^T x) \text{ s.t.}$$

$$\begin{cases} Ax \succeq b \\ x \succeq 0 \end{cases}$$

There exists an efficient algorithm to solve a linear programming.

- Simplex method : non-polynomial time, practical method
- Elliposoid method : polynomial-time in theory, not very good in practice.
- Interior Point method : good at both side.

Integer Programming :

$$\min(c^T x) \text{ s.t.}$$

$$\begin{cases} Ax \succeq b \\ x \in \{0,1\}^n \iff x_i \in \{0,1\}, \forall i \in [n] \end{cases}$$

Integer Programming is NP-hard problem.

7.2 $(1-\frac{1}{e})$-Approximation Algorithm for maximum Coverage

Given this $S_1, S_2, \cdots, S_m \subseteq [n]$
$k \geq 1$
Output : $I \subseteq [m], |I| \leq k$
maximize $|\cup_{i \in I} S_i|$
we can take this problem as Integer program by using the following expression:

$y_i : i \in [m]$, whether we take S_i or not, $y_i \in [0,1]$
$x_j \in 0, 1, j \in [n]$, whether j is covered or not.
Output : max $\sum_j x_j$
We can observe two relationship from this problem:

$$x_j \leq \sum_{i \in [m], j \in S_i} y_i, \forall j \in [n]$$

$$\sum_{i=1}^{m} y_i \leq k$$

$$x_j \in 0, 1, \forall j \in [n]$$

$$y_i \in 0, 1, \forall i \in [m]$$

This is a NP-hard probelm and we can't solve it efficiently, but we can convert this Integer Program to Linear Program by making the following changes.

LP relaxation for max coverage :

$$max \sum_j x_j \quad \text{s.t.}$$

$$x_j \leq \sum_{i \in [m], j \in S_i} y_i, \forall j \in [n]$$

$$\sum_{i=1}^{m} y_i \leq k$$

$$x_i \in [0,1], \forall j \in [n]$$

$$y_i \in [0,1], \forall i \in [m]$$

let :

$$\text{opt} = \text{value of IP}$$
$$\text{lp} = \text{calue of LP}$$

Goal :

$sol \geq (1 - \frac{1}{e}) \cdot lp \geq (1 - \frac{1}{e}) \cdot opt$

y_i in IP : indicating whether S_i is selected or not
y_i in LP : can be viewed as the probability that we selecting S_i

---

Algorithm 1 Rounding Algortihm 1

---

for each $i \in [m]$ **do**
  w.p. y_i let $\widetilde{y_i} = 1$
  w.p. $1 - y_i$ let $\widetilde{y_i} = 0$
for each $j \in [n]$ **do**
  let $\widetilde{x_j} = \min\{1, \sum_{i:j \in S_i} \widetilde{y_i}\}$
return $\{S_i : \widetilde{y_i} = 1\}$

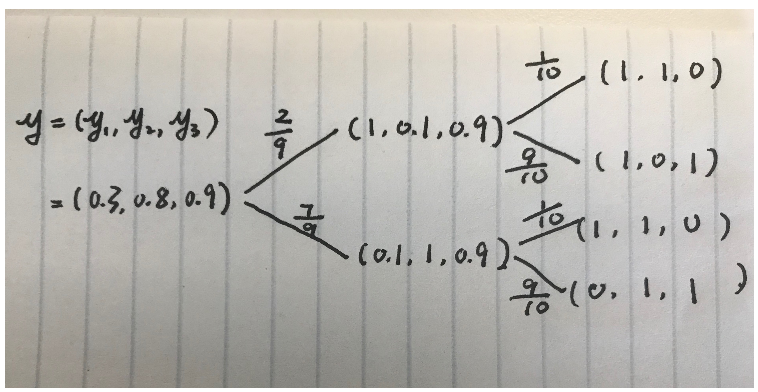

Figure 7.1: Figure of example of Dependence rounding

Dependence Rounding : In every iteration, make one more coordinate integral and keep the marginal probabilities until get a integral vector. Above is a figure of dependence rounding :

$$y = (y_1, y_2, y_3) = (0.3, 0.8, 0.9)$$

$$\widetilde{y} \in \{0,1\}^3, \quad \text{s.t.} \widetilde{y_1} + \widetilde{y_2} + \widetilde{y_3} = 2$$

$$\mathbb{E}\,\widetilde{y} = y$$

The number on each edge in the figure is the marginal probability.

Lemma 7.1

$$\mathbb{E}[\widetilde{x_j}] \geq (1 - \frac{1}{e}) x_j$$

Algorithm 2 Rounding Algortihm 2

$\widetilde{y_i} = 0, \forall i \in m$
for $t \leftarrow 1$ to k **do**
    choose i^* accroding to the following distribution:
    $Pr[i^* = i] = \frac{y_i}{\sum_{i=1}^m y_i}$
    let $\widetilde{y_i} \leftarrow \widetilde{y_i} + 1$
return $\{S_i : \widetilde{y_1} \geq 1\}$
for each $j \in [n]$ **do**
    $\widetilde{x_j} = \min\{1, \sum_{i:j \in S_i} \widetilde{y_i}\}$

$$\Rightarrow \mathbb{E}[\sum_{j=1}^n \widetilde{x_j}] \geq (1 - \frac{1}{e} \sum_{j=1}^n x_j)$$

Proof:

$$\begin{aligned} \mathbb{E}[\widetilde{x_j}] = 1 - (1 - \frac{\sum_{i:j \in S_i} y_i}{\sum_{i=1}^m y_i})^k &\geq 1 - (1 - \frac{\sum_{i:j \in S_i} y_i}{k})^k \\ &\geq 1 - (1 - \frac{x_j}{k})^k \\ &\geq 1 - [e^{-x_j/k}]^k = 1 - e^{(-x_j)} \end{aligned}$$

■

Note : $1 - x \leq e^{-x}$, true $\forall x$

Need :

$$\frac{1 - e^{-x_j}}{x_j} \geq 1 - \frac{1}{e} \quad \text{for} \quad x_j \in (0, 1]$$