大数据算法-2025 春

Lecture 3: 随机算法——Maxcut

2025.3.6

Lecturer: 丁虎 *Scribe:* 沈俊杰

1 一些前置知识

下面介绍一些本篇讲义中需要用到的知识及定义。

Definition 1.1. 随机算法是一种在算法过程中引入随机函数，且随机函数的返回值直接或间接影响了算法的执行流程或执行结果。一般的随机算法分类有以下几种：

- Las Vegas 算法：算法总是返回正确的结果。（但不保证返回结果）
- Monte Carlo 算法：算法返回的结果未必正确，只能保证正确（错误）的概率。
- Sherwood 算法：消除输入样例对计算复杂度的影响。

Definition 1.2. 凸优化问题。

一般优化问题一般优化问题的形式为

$$\begin{aligned} \min \quad & f_0(x) \\ \text{s.t.} \quad & f_i(x) \le 0, \quad i = 1, \dots, m \\ & h_i(x) = 0, \quad i = 1, \dots, n \end{aligned}$$

注意：凸优化问题要求：

- $f_0(x)$ 是下凸函数
- 可行域（Feasible Domain），即满足约束的 x 的范围，是一个凸域。

在机器学习等任务中，凸优化问题非常常见，一些经典的方法包括梯度下降被广泛应用。

Example 1.3. *SDP problem:*

$$
\begin{aligned}
\text{变量：}\quad & \{x_{ij} | 1 \leq i, j \leq n\} \\
\text{目标：}\quad & \boldsymbol{min} \sum_{i,j} \alpha_{ij} x_{ij} \\
\text{约束：}\quad & \mathbb{X} = (x_{ij}) \quad \text{半正定} \\
& \cdots \quad \textit{(一些关于}x_{ij}\textit{的线性不等式组)}
\end{aligned}
$$

Remark 1.4. SDP 问题是凸优化问题，因为变量空间 $\mathbb{X}$ 是一个锥（cone），特殊的，这类问题可以叫做锥优化。

Definition 1.5. 整数线性规划:

一类特殊的线性规划，变量取值仅可为整数值。更为特殊的，仅可为 0-1 值，此时称为 0-1 规划。

2 Max Cut

接下来我们考虑一个具体的问题——最大割问题。

Definition 2.1. Max Cut：

$$
\begin{aligned}
Input: \quad & G = (V, E) \quad , V = \{1, 2, \cdots, n\} \quad \forall i, j \in V, \quad w_{ij} \geq 0 \\
Output: \quad & S = \arg\max_{S \subseteq V} w(S, V \backslash S), \quad w(S, V \backslash S) = \sum_{i \in S, j \notin S} w_{ij}
\end{aligned}
$$

我们使用 SDP 解决这个问题。首先转化一下问题:

$$
\sum_{i \in S, j \notin S} w_{ij} = \frac{1}{2} \sum_{i \leq j} w_{ij} (1 - y_i y_j) \tag{1}
$$

$$
y_i = \begin{cases} +1, & i \in S \\ -1, & i \notin S \end{cases} \tag{2}
$$

我们进一步做 Relaxation：

$$
y_i = \{\pm 1\}_n \Longrightarrow v_i \in \mathbb{S}^n \tag{3}
$$

Remark 2.2. $\mathbb{S}^n$是$\mathbb{R}^n$上的单位球面, 原先的 $y_i = \{\pm 1\}$ 是 $\mathbb{S}^1$, 后者是前者的特殊情况。注意，这样的 Relaxation 会导致目标函数的最优值偏大（可行域变大了）。我们记此时的 Problem 为 $\mathbb{B}$，区别于原始的问题 $\mathbb{A}$

将缩放之后的形式带回（1）中，我们令 $x_{ij} =< v_i, v_j >$

$$\frac{1}{2}\sum_{i\leq j} w_{ij}(1-y_iy_j) = \frac{1}{2}\sum_{i\leq j} w_{ij}(1-x_{ij}) \tag{4}$$

可以看出，此时的目标函数形式已经是 SPD 形式，我们来证明 $\mathbb{X}$ 是半正定的。

Proof.

$$\begin{aligned} y^T\mathbb{X}y &= \sum_{ij} y_iy_jx_ij = \sum_{ij} y_iy_j < v_i, v_j > && (5)\\ &= \sum_{ij} < y_iv_i, y_jv_j >=< y_1v_1 + y_2v_2 + \cdots + y_nv_n, y_1v_1 + y_2v_2 + \cdots + y_nv_n > && (6)\\ &= ||y_1v_1 + y_2v_2 + \cdots + y_nv_n||^2 \geq 0 && (7) \end{aligned}$$

□

Remark 2.3. 得到 $\mathbb{X}$，如何得到最初始的 $\{v_1, \cdots, v_n\}$?
事实上，$\mathbb{X}$ 和 $\{v_1, \cdots, v_n\}$ 相差一个旋转因子，不过这对问题的性质没有关系。对于正定矩阵，我们可以使用 Cholesky 分解，即 $A = L^TL$，式中 L 是下三角阵，每行是一个 v_i。

完整的 Max Cut 算法如下:

1. 通过上述的转化，将目标函数转化为 (4) 的形式，此时问题是一个 SDP 问题。求解得到 $\mathbb{U} = \{v_1, \cdots, v_n\} \subseteq \mathbb{S}^n$
2. 在 $\mathbb{S}^n$ 上随机选取一个向量 r, 并根据 r 将 $\mathbb{U}$ 分类:

$$S = \{i| < v_i, r >\geq 0\}$$
$$V\backslash S = \{i| < v_i, r >< 0\}$$

3. $\{S, V\backslash S\}$ 是一个 Max Cut 的解。

3 算法分析

我们尝试分析算法的近似效果。

$$
\begin{aligned}
\mathbb{E}[w(S, V\backslash S)] &= \sum_{i<j} w_{ij} \cdot Prob[sgn(<v_i, r>) \neq sgn(<v_j, r>)] && (8)\\
&= \sum_{i<j} w_{ij} \frac{2\text{arccos}(<v_i, v_j>)}{2\pi} && (9)\\
&= \frac{1}{\pi}\sum_{i<j} w_{ij}\text{arccos}(<v_i, v_j>) = \Delta && (10)
\end{aligned}
$$

Remark 3.1. 请注意，$\{v_1, v_2, \cdots, v_n\}$ 仍然是在 $\mathbb{S}^n$, 而不是 $\mathbb{S}^2$ 上，但考虑到两两向量之间时，其组成的空间是 2 维的。

再考虑最优值 OPT 的一个上界 U，得到我们结果的近似比 $\alpha = \frac{\Delta}{OPT} \geq \frac{\Delta}{U}$。由于问题 $\mathbb{B}$ 是由问题 $\mathbb{A}$ 放松得到的，也就是说问题 $\mathbb{B}$ 的最优解一定大于 $\mathbb{A}$ 的最优解。

问题 $\mathbb{B}$ 的目标函数是 $\frac{1}{2}\sum_{i\leq j} w_{ij}(1- <v_i, v_j>)$, 那么我们可以推出近似比。

$$
\begin{aligned}
\alpha &= \frac{2}{\pi}\frac{\sum_{i<j} w_{ij}\text{arccos}(<v_i, v_j>)}{\sum_{i\leq j} w_{ij}(1- <v_i, v_j>)} && (11)\\
&\geq \frac{2}{\pi}\min\frac{\text{arccos}(<v_i, v_j>)}{(1- <v_i, v_j>)} && (12)\\
&\geq \frac{2}{\pi}\min\frac{\theta}{1-\cos\theta} \approx 0.878 && (13)
\end{aligned}
$$

根据之前的 Markov 不等式，我们还能得到一些结果。

$$
\begin{aligned}
\mathbb{E}[\frac{OPT-\Delta}{OPT}] &< 0.122 \Longrightarrow && (14)\\
\text{Prob}[\frac{OPT-\Delta}{OPT} > 0.122(1+\epsilon)] &< \frac{\mathbb{E}[\frac{OPT-\Delta}{OPT}]}{0.122(1+\epsilon)} && (15)\\
&< \frac{1}{1+\epsilon} && (16)\\
& && (17)
\end{aligned}
$$

通过 Lecture1 中的 Uninon Bound，可以通过多次重复，提高概率，这也是随机算法中常见的操作。

4 补充

- Max Cut 问题是 Apx-Hard 问题，不存在现有的 PTAS 算法。且对于近似比 >0.941 的情况均为 NPH 问题
- 如果 Unique Games Conjecture 正确，那么 0.878 已经是最好的近似比。UG 猜想与本章的关系在于，如果其成立，那么 SDP 可以为一大类的近似问题提供**最优**的近似比，其中也包含 Max Cut。其提出者为 Subhash Khot[1]。可以参考的一些 note: stanford Khot

References

[1] S. Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 767–775, 2002.