浙江大学
ZHEJIANG UNIVERSITY

College of Computer Science and Technology

计算机科学与技术学院

Final Report

数据要素市场大作业报告

Author:

李瀚轩，郑涵文，王垠凯

Report topic:

利润最大化数据定价

2024 年 7 月 31 日

目录

第 1 章 Abstract

定价问题是经济学中的重要议题，然而消费者偏好的多样性和市场信息的不对称性给定价带来了新的挑战。本篇报告中，我们调研了三篇采用现代算法理论的文章来尝试解决利润最大化数据定价这一问题：

1. **Approximation Algorithms for Single-minded Envy-free Profit-maximization Problems with Limited Supply**.
   这篇文章首次提出了针对有限供应下单一意向消费者无嫉妒利润最大化问题的多项式时间近似算法。文章将社会福利最大化问题与利润最大化问题相结合，通过算法分析提供了一种新的解决方案，以实现在有限供应条件下的最优利润。

2. **On Profit-Maximizing Envy-free Pricing**
   这篇文章较之上一篇讨论问题深度更浅但范围较广，对单需求消费者无嫉妒利润最大化问题和单一意向无嫉妒利润最大化问题进行了研究，对前者在有限供应条件下提出了对数级别近似算法、对后者在无限供应条件下提出了对数级别近似算法。此外，对于这两类问题的一些特殊情况，给出了多项式时间算法或伪多项式时间算法。

3. **Single-Minded Unlimited Supply Pricing on Sparse Instances**
   这篇文章主要针对前文中提到的无限供应下单一意向无嫉妒利润最大化问题进行了深入研究，证明了该问题是 APX-hard 的，并针对一般定价问题提出了两种近似算法。

本篇报告首先在第二章对单一意向消费者无嫉妒利润最大化问题（SMEFP）展开了介绍和研究，使用线性规划方法给出了一个有限供应情况下的多项式时间近似算法，随后给出了一个简单的无限供应情况下的对数级别近似算法。随后讨论了收费站问题（一类 SMEFP 问题）的特殊情况下的多项式或伪多项式算法。

第三章使用了另一种等价定义，对 SMEFP 问题在无限供应的情况下进行了更深入的讨论，给出了两个较上一章更为复杂的近似算法；同时对第二章中收费站问题的几个特殊情况进行了更详细的分析。

第四章讨论了单需求消费者无嫉妒利润最大化问题（UDEFP），对有限供应条件下提出了对数级别的近似算法，同时讨论了随时间定价问题（UDEFP 的一类特殊情况），提出了无限供应条件下的多项式复杂度算法和有限供应条件下的伪多项式复杂度算法。最后，报告提出了一个诚实的竞争拍卖机制用于消费者单需求的情况，该机制可以做到对数级别的近似。

第 2 章 Single-minded Envy-free Problems

2.1 问题定义

我们首先给出单一意向无羡慕利润最大化问题 (Signle-minded Envy-free Profit-Maximization Problems，下文用 SMEFP 代替) 的定义。首先我们对该问题的一些基本概念进行解释：

1. 单一意向消费者：每个消费者只对一组特定的商品感兴趣，并愿意为这组商品支付一定的价值。

2. 无羡慕解决方案：每个顾客对自己的分配感到满意，直观地说，如果一个消费者没有获得他想要的商品，那么他对于获得该商品所需支付的价格应该高于他内心的估值，这样他就不会羡慕那些获得商品的消费者。

此外，SMEFP 还涉及到商品供应量的问题，分为有限供应和无限供应两种情况。本节会对两种情况各提出近似算法。

对这个问题有了直观理解之后，我们可以给出 SMEFP 问题的形式化定义：

定义 1.

Input:

- 商品集合：m 个可供出售的商品，记作 $E = \{e_1, e_2, \cdots, e_m\}$。
- 顾客集合与需求：n 个顾客，每个顾客 i 有一个单一意向的商品子集 $S_i \subseteq E$, 即顾客 i 只对这组特定的商品感兴趣。
- 顾客估值：每个顾客 i 有一个内心估值 v_i, 表示他们为获得 S_i 愿意支付的最大金额。
- 商品供应量: 每个商品 e 有 u_e 个供应量（无限供应时 $u_e = +\infty$）。

Output: 算法为每个商品 e 确定一个非负价格 p_e，以及给出获胜者集合 $W \subseteq \{i : \sum_{e \in S_i} p_e \leq v_i\}$，同时满足以下约束条件：

- 容量约束：$|\{i \in W : e \in S_i\}| \leq u_e$, 即每个商品 e 只能分配给最多 u_e 个获胜者。
- 无羡慕约束：对于 $i \notin W$ 的每个顾客, 有 $\sum_{e \in S_i} p_e \geq v_i$，即没有顾客会羡慕其他顾客的分配。

2.2 Preliminaries

在本小节，我们介绍本篇文章的算法需要用到的一些基本概念。

2.2.1 social-welfare maximization

福利最大化问题是指在给定的商品集合和顾客需求下，如何确定商品的价格以最大化获胜者集合的总估值。

2.2.2 Linear Programming Basics

给定如下一个线性规划问题, 记作 P：

$$\begin{aligned} &\max\ c^T x \\ &s.t.\ Ax \le b \\ &x \ge 0 \end{aligned} \tag{2.1}$$

那么我们可以得到其对偶问题, 记作 D：

$$\begin{aligned} &\min\ b^T y \\ &s.t.\ A^T y \ge c \\ &y \ge 0 \end{aligned} \tag{2.2}$$

可以观察到，对偶问题中引入了与原问题约束相对应的变量，并且原问题中的约束系数在对偶问题中成为目标函数的系数。接下来我们给出与本篇文章的算法设计相关的对偶问题的几个定理：

定理 1. 弱对偶定理：*假设 $\boldsymbol{x}, \boldsymbol{y}$ 分别是 P, D 的可行解，那么我们有 $c^T x \le b^T y$。*

定理 2. 互补松弛定理: *设 x^*, y^* 分别是 P, D 的可行解，那么 x^*, y^* 是 P, D 的最优解当且仅当:*

$$\begin{cases} (y^{*T} A - c^T) x^* = 0 \\ y^{*T} (A x^* - b) = 0 \end{cases} \tag{2.3}$$

通过互补松弛定律我们可以得到一个重要的结论：在线性规划的最优解中，如果一个变量的值为正，则与之相对应的对偶约束必须是紧的；如果一个约束是宽松的，那么对应的对偶变量在最优解中的值为零。这个结论在后续的算法设计中将会被频繁使用。

2.3 有限供应下 SMEFP 问题的近似算法

在本节中，我们将设计一个多项式时间的近似算法，以解决 SMEFP 问题。

2.3.1 用对偶理论描述问题

我们首先介绍解决 SMEFP 问题的一个关键思想：使用线性规划及其对偶问题来确定商品价格和获胜者集合。

给定 SMEFP 问题的输入供应量 (u_e), 顾客需求 $\{S_i\}$, 估值 $\{v_i\}$，我们首先考虑较为简单的 SWM 问题：我们引入变量 x_i，如果顾客 i 被选作赢家，则 $x_i = 1$, 否则 $x_i = 0$。显然我们可以把 SWM 问题描述为一个整数规划问题，但我们首先将其作 LP-松弛，转化为一个一般的线性规划问题:

$$\begin{aligned} &\max \sum_i v_i x_i \\ &\textbf{s.t.} \sum_{i:e\in S_i} x_i \le u_e, \textbf{for all } e \\ &0 \le x_i \le 1, \textbf{for all } i \end{aligned} \tag{2.4}$$

我们将上述问题记作 P，可以得到，P 是满足容量约束的条件的。但是我们还需要满足无羡慕的约束，因此我们考虑 P 的对偶问题 D:

$$\begin{aligned} &\min \sum_e u_e y_e + \sum_i z_i \\ &\textbf{s.t.} \sum_{e\in S_i} y_e + z_i \ge v_i, \textbf{for all } i \\ &y_e \ge 0, z_i \ge 0, \textbf{for all } e, i \end{aligned} \tag{2.5}$$

这时我们考虑对偶问题 D 的最优解 (y^*, z^*)，其中 y_i^* 代表每件物品的价格。根据互补松弛定理，我们可以得到以下结论：

1. 如果参与人 i 是获胜者，此时 $x_i^* > 0$, 那么 $\sum_{e\in S_i} y_e^* + z_i^* = v_i$，由于 $z_i^* \ge 0$, 因此 $\sum_{e\in S_i} y_i^* \le v_i$。
2. 如果参与人 i 不是获胜者，即 $x_i^* < 1$, 那么 $z_i^* = 0$, 因此 $\sum_{e\in S_i} y_i^* \ge v_i$。

因此我们可以发现，由原问题 P 和对偶问题 D 得到的获胜者集合 $\{x_i^*\}$ 和定价策略 $\{y_e^*\}$ 满足无羡慕的约束条件。

这里我们需要提前给出一个引理，该引理在后续的近似比证明中起到作用。

引理 1. *设 D 问题的最优解为 OPT, 对于供应量 k，我们有 $OPT(k)$ 是关于 k 的分段线性凹函数。进一步地，对于 $\lambda \in [0,1]$，那么若 $OPT(\cdot)$ 在 k_0 和 k_0' 之间的线段上是线性的当且仅当存在一个对偶解 (y,z)，它同时是 (D_{k_0}) 和 $(D_{k_0'})$ 的最优解。*

2.3.2 近似算法设计思路

在上一小节中，我们将 SMEFP 问题转化了一个线性规划问题 (P) 以及其对偶问题 (D)，我们可以通过求解 P 和 D 来得到问题的一个可行解。但是需要注意的是，上述转化并不完全适配 SMEFP 的所有情况，并且所得到的最优解也不一定是原问题的最优解，具体表现在以下两个方面：

1. 由于 P 是一个经过 LP 松弛之后的问题，因此其最优解可能不是整数。然而在上一节的设定下，我们给出的解是整数解 ($x_i = 1$ 或 $x_i = 0$)，即我们在松弛条件的设定下对解强加了整数要求，这可能会导致解的不准确。

2. D 问题的最优解可能也不是 SMEFP 的最优解。

接下来我们首先放宽问题的条件，将由简单到复杂，详细探讨解决上述两个问题的思路，最终得到 SMEFP 的一个近似算法。

2.3.3 Uniform Supply + P Integer Optimal Case

我们首先放宽条件，考虑一个简化的情况，即所有的商品容量都是一样的，记作 U，并且假设 P 的最优解就是整数解。因此我们可以从 P 问题的整数最优解，即获胜者集合出发，结合其对偶问题的最优解，即价格集合，通过互补松弛定理可以得到简化后 SMEFP 问题的一个可行解：

引理 2. *令 $k \le U$, 假设 x^* 是 P_k 问题的整数最优解，(y^*, z^*) 是 D_k 的最优解，那么 $(y^*, W = \{i : x_i^* = 1\})$ 是简化后问题的一个可行解，其中利润 $\sum_e u_e y_e^* = \sum_e k y_e^*$.*

这里实际上就解决了 2.3.2 中的第一个问题。而对于第二个问题，原文中的想法是通过减少边的容量尽可能获得更高的利润，我的理解就是遍历所有满足 $k \le U$ 的 k, 找到最大化利润的对偶问题的解，然后通过这个解得到获胜者集合，同时对容量的改变是有一个近似比的范围保证的。下面给出这个算法：

Algorithm 1

1. 对于 $k = 1, \cdots, U$, 计算 (D_k) 和 (D_{k-1}) 的最优解 $(y^{(k)}, z^{(k)})$。

2. 选择 $c \in \{1, \cdots, U\}$ 使得 $\sum_e u_e y_e^{(c)}$ 最大，并计算 (P_c) 的最优解 $x^{(c)}$。

3. 返回定价方案 $y^{(c)}$ 和获胜者集合 $W = \{i : x_i^{(c)} = 1\}$。

但是注意到我们逐个考虑 $[1, U]$ 的每个 k 比较慢，我们采用 $(1+\epsilon)$ 的步长，设置 $k_{j+1} = \lceil (1+\epsilon) k_j \rceil$。或许我们会疑惑在算法的步骤 1 中我们为什么要取 D_k 和 D_{k-1} 的共同最优解。我们首先看一下近似比的证明过程：

定理 3. *算法返回的解至少能够达到 $\dfrac{OPT(U)}{H_U}$ 的利润，其中 H_U 是调和级数。*

证明. 令 $P = \sum_e c y_e^{(c)} = \max_{k-1 \cdot U} \sum_e k y_e^k$, 这是通过算法得到的结果。注意到对于 $k = 1, \cdots, U$, 我们有 $\dfrac{P}{k} \ge \sum_e y_e^{(k)} = OPT(k) - OPT(k-1)$。将所有项相加即可得到 $P \cdot H_U \ge OPT(U)$. □

观察证明过程，我们可以发现，由引理 1，D_k 和 D_{k-1} 的共同最优解保证了 OPT 的分段线性性，换个角度想，OPT 的间断点都是整数点，而整数断点的存在允许算法 • 1 在不同的容量水平上选择最优的 y 值，因此能够进行 telescope sum 的求和。这也解答了我们的疑惑，并且这个性质在更一般的情况下也起着同样的作用。
由该定理我们可以很自然地得到以 $(1+\epsilon)$ 的步长更新 k 的算法的近似比是 $(1+\epsilon)H_U$。到这里我们已经解决了这个最特殊的情况，即我们假设 P 问题有整数最优解，并且所有商品供应量是相同的。接下来我们逐个放宽条件。

2.3.4 Uniform Supply Case

首先我们去掉 P 问题具有整数最优解的约束，给出一个 LP-rounding 的算法，使得我们可以将分数解转化成我们已经熟悉的整数解。首先我们给出一个引理。

引理 3. *假设 $k=(k_e)$ 是商品供应量，x 是 P_k 问题的分数解，$\mathcal{A}$ 是 SWM 问题基于线性规划的 α 近似的多项式时间算法，那么在多项式时间内，我们可以将 $\frac{x}{\alpha}$ 表示为 (P_k) 整数解的凸组合。*

因此我们很直观的想法就是将分数解表示成整数最优解的凸组合，将问题转化为上一小节我们介绍的情况。下面我们给出这种情况下的算法以及 LP-rounding 的过程。
Round$(\mu=(\mu_e), x^*)$　首先定义 $\mu_e^{'}=\mu_e-|\{i:x_i^*=1, e\in S_i\}|$, 即剩余的商品供应量。我们用新的供应量 $\mu^{'}$ 定义一个新的 LP 问题 $(P_{\mu'})$，并且如果 $x_i^*<1$ 就令 $x_i=x_i^*$, 否则 $x_i=0$。然后利用凸分解定理将 $\frac{x}{\alpha}$ 分解成整数解 $\hat{x}^k$ 的凸组合，即 $x=\sum_{k=1}^{l}\lambda_k\hat{x}^k$。最后采用随机算法返回获胜者集合 W: 以概率 λ_k 选择 $\hat{x}^k$ 作为获胜者集合，并将 W 设置为 $\{i:x_i^*=1\}\cup\{i:\hat{x_i}^k=1\}$。
Algorithm 2　给定具有相同供应量 $u_e=U$ 的 SMEFP 问题的一个输入实例，以及 LP-Based 的一个 SWM 问题的 α 近似算法：

1. 对于每个 $k=k_1,\cdots,k_l$, 其中 $k_j=(1+\epsilon)k_{j-1}$，得到每个 (D_k) 的最优解，并在这其中选出最大化 $\sum_e ky_e^*$ 的解 $(y^{(k)}, z^{(k)})$。
2. 选择使得 $\sum_e cy_e^{(c)}$ 最大的 $c\in\{k_1,\cdots,k_l\}$，并计算 (P_c) 的最优解 $x^{(c)}$。
3. 返回定价策略 $y^{(c)}$, 获胜者集合 $Round(c, x^{(c)})$.

通过 2.3.1 中的类似思路，我们可以证明算法返回的解是 SMEFP 问题的一个可行解, 并且选择 winner set 的随机算法对期望利润有个 α 近似比的保证。现在我们为了证明算法第二步中选择的 c 相对于 OPT 是比较大的 (这样子能够保证近似比的范围) 我们首先需要一个表达式来描述 $\sum_e ky_e^{(k)}$ 的可能值，并且这个描述的性质类似一般的分段线性性，便于我们在近似比的证明中使用。下面首先给出这个引理。

引理 4. *对任意 k, 我们有 $\sum_e k y_e^{(k)} = k \cdot \dfrac{OPT(k) - OPT(b_k)}{k - b_k}$。其中 b_k 是使得 $OPT(\cdot)$ 在 $[b_k, k]$ 区间内线性的最小值 (即 k 之前的最近断点)。*

至此我们可以给出在 Uniform Supply 条件下的算法近似比。

定理 4. *对任意的 $\epsilon > 0$, 算法 2 在 **poly(input size, $\frac{1}{\epsilon}$)** 的多项式时间内返回一个近似比为 $(2\alpha(1+\epsilon)H_U)$ 的解。*

2.3.5 Arbitrary capacities case

现在我们可以给出最一般情况的 SMEFP 问题的近似算法。注意到在没有了相同供应量的限制之后，我们唯一需要改变的就是算法输入的供应量，其从原先的常数值变成一个向量。

Algorithm 3 给定 SMEFP 问题的实例以及 LP-Based 的一个 SWM 问题的 α 近似算法：

1. 我们按如下方式定义 $k^1, k^2, \cdots, k^l$: 对于所有商品 e, $k_e^1 = 1$。对于 $j > 1$, 定义 $k_e^j = \min\{\lceil (1+\epsilon) k_e^{l-1}, u_e\}$。对于每个供应量向量 $k = k^j, j = 1 \cdots, l$, 计算每个 (D_k) 的最优解，并在这其中选出最大化 $\sum_e k y_e^*$ 的解 $(y^{(k)}, z^{(k)})$。

2. 选择使得最大化 $\sum_e c y_e^{(c)}$ 的向量向量 $c \in \{k^1, \cdots, k^l\}$，并计算 (P_c) 的最优解 $x^{(c)}$。

3. 返回定价策略 $y^{(c)}$, 获胜者集合 $Round(c, x^{(c)})$。

首先，算法依然能够确保返回的解是 SMEFP 问题的一个可行解，并且随机算法的利润保证依然成立。最终版本算法对于近似比分析的困难之处在于，现在的 $OPT(\cdot)$ 是一个多变量函数，我们对函数间断点的分析也要扩展到特定的方向。这里我们定义 $b_{k,d}$ 其中 k 是供应量向量，$d \in \mathbb{R}^m_{\geq 0}$ 是一个向量，$b_{k,d}$ 是使得 $OPT(\cdot)$ 在连接 k 和 $k - d(1-r)$ 的线段上是线性的，其中 $r \geq 0$ 并且 $r \in [\max_e(1 - \frac{k_e}{d_e}), 1]$。即 $b_{k,d}$ 是沿方向 d 在 k 之前的断点。对于方向向量 d, 我们定义 $d_j = k_j - k_{j-1}$。
现在我们有了间断点的定义，follow 之前特殊情况的思路，我们可以给出一个引理来描述 $\sum_e k y_e^{(k)}$。

引理 5. *对于任意供应量向量 k 和向量 $d \in \mathbb{R}^m_{\geq 0}$ $d < k$，我们有:*

$$\left(\sum_e k_e y_e^{(k)}\right) \cdot \max_e \frac{d_e}{k_e} \geq \frac{OPT(k) - OPT(k - d(1 - b_{k,d}))}{1 - b_{k,d}} \tag{2.6}$$

当 $d = k$ 时等号成立。

因此我们可以证明算法 3 的第二步中选取的 c 的利润相对于 OPT 有个下界的保证。

引理 6. 算法选取的 c 满足

$$\sum_e c_e y_e^{(c)} \geq \frac{OPT(u)}{(2(1+\epsilon)H_{u_{\max}})} \tag{2.7}$$

证明. 我们令 e^* 是具有最大供应量的商品，即 $u_{e^*} = u_{\max}$。
设 $P = \sum_e c_e y_e^{(c)} = \max_k \sum_e k_e y_e^{(k)}$。我们首先固定 $j < l$，定义 $k_0 = k^{j+1} - d^{j+1}(1 - b_{k^{j+1},d^{j+1}})$。
由于我们有 $\frac{d_{e^*}^j}{k_{e^*}^j} = \max_e(\frac{d_e^j}{k_e^j})$，将其代入引理 5 中的不等式，可以得到：

$$P \cdot \frac{d_{e^*}^{j+1}}{k_{e^*}^{j+1}} \geq \frac{OPT(k^{j+1}) - OPT(k')}{1 - b_{k^{j+1},d^{j+1}}} \tag{2.8}$$

又由于 $\frac{d_{e^*}^j}{k_{e^*}^{j-1}} = \max_e(\frac{d_e^j}{k_e^{j-1}})$, 同样地我们有:

$$P \cdot \frac{d_{e^*}^{j+1}}{k_{e^*}^{j}} \geq \frac{OPT(k') - OPT(k^j)}{b_{k^{j+1},d^{j+1}}} \tag{2.9}$$

将 (2.8) 式乘上 $a = b_{k^{j+1},d^{j+1}}$ ，(2.9) 式乘上 $1-a$, 我们可以得到:

$$P \cdot [(1-a) \cdot \frac{d_{e^*}^{j+1}}{k_{e^*}^{j+1}} + a \cdot \frac{d_{e^*}^{j+1}}{k_{e^*}^{j}}] \geq OPT(k^{j+1}) - OPT(k^j) \tag{2.10}$$

由于我们选取 k 的步长，我们有 $k_{e^*}^{j+1} \leq 2k_{e^*}^j(1+\epsilon)$，因此我们可以得到:

$$P \cdot [(1-a) \cdot \frac{d_{e^*}^{j+1}}{k_{e^*}^{j+1}} + a \cdot \frac{d_{e^*}^{j+1}}{k_{e^*}^{j}}] \leq 2(1+\epsilon) \cdot P \cdot \sum_{t=k_{e^*}^j+1}^{k_{e^*}^{j+1}} \frac{1}{t} \tag{2.11}$$

即:

$$2(1+\epsilon) \cdot P \cdot \sum_{t=k_{e^*}^j+1}^{k_{e^*}^{j+1}} \frac{1}{t} \geq OPT(k^{j+1}) - OPT(k^j) \tag{2.12}$$

将 (2.12) 式各项相加，即可得到 $P \cdot 2(1+\epsilon)H_{u_{\max}} \geq OPT(u)$。 □

至此，结合在 **Round** 步骤中随机算法的 α 近似比的利润期望保证，我们可以给出一般条件下 SMEFP 问题的近似比。

定理 5. 算法 3 可以在多项式时间内返回一个近似比为 $(2\alpha(1+\epsilon)H_{u_{\max}})$ 的解。

至此，我们完成了有限供应下 SMEFP 问题多项式时间近似算法的设计与分析。

2.4 无限供应下 SMEFP 问题的一个简单近似算法

无限供应显然比有限供应更容易一些，在这里我们给出一个简单的对数近似算法。我们有更复杂的近似算法，将在下一章详细分析。

algorithm 4

1. 对每个消费者 i 计算 $q_i = \frac{v_i}{|S_i|}$。

2. 固定 i，将所有物品定价为 q_i，计算此时的利润。对所有 i 均做一次该步骤。

3. 选择能够得到最大利润的 q_i 作为所有物品的统一价格。

可以看到这个算法思路非常简单而暴力，给所有的物品统一定价，但是具有对数级别的近似比。

定理 6. *上述算法具有 $O(\log n + \log m)$ 的近似比。*

证明. 假设消费者按 $q_1 \geq q_2 \geq \cdots \geq q_n$ 排序。如果所有物品价格为 q_i，则卖家利润 $R_i = \sum\limits_{1\leq j\leq i} |S_j| \cdot \frac{v_i}{|S_i|}$。

也就是 $v_i = |S_i| \dfrac{R_i}{\sum\limits_{1\leq j\leq i} |S_j|}$。

算法选择了最大化利润的 R，即 $R_i \leq R$ 对所有 i 成立。因此

$$\sum_{i=1}^{n} v_i = \sum_{i=1}^{n} |S_i| \frac{R_i}{\sum\limits_{1\leq j\leq i} |S_j|} \leq R \cdot \sum_{i=1}^{n} \sum_{k=1}^{|S_i|} \frac{1}{k + \sum\limits_{1\leq j\leq i-1} |S_j|} \leq R \cdot \sum_{i=1}^{n} \ln(|S_i|) \leq R \cdot \ln(\sum_{i} |S_i|)$$

而 $\sum\limits_{i=1}^{n} v_i$ 显然是最优界的上界，且 $\sum\limits_{i} |S_i| \leq nm$，因此定理得证。 □

2.5 收费站问题

收费站问题是 SMEFP 问题的另一种形式，在这里物品是图 G 上的边，我们可以将其视为公路段，客户的请求是图中的路径，会对某条路径有一个估值。卖家是高速公路系统的所有者，希望为每条边选择收费以最大化利润。本节将会讨论收费站问题的几个特例，并提出多项式或伪多项式算法。下一章将会证明收费站问题是 APX-hard 的，并对几个特例进行进一步的详细分析。

2.5.1 树上收费站

一个特例是，当所有消费者的路径请求共享一个公共端点 r，且这些路径构成了一棵树时，我们称之为树上收费站问题。这时可以把 r 看作树根，在树上使用动态规划算法。

首先讨论无限供应的情况，即边的容量无限。

我们给出一个动态规划算法。

对于节点 w，定义 R_w 表示所有起始于以 w 为根的子树 T_w 中的请求集合。定义 $a(w,b)$ 为在从 w 到根路径的成本恰好为 b 时，从 R_w 中获得的最优收入。我们关注计算 $a(r,0)$。

对于节点 w 记 $w_1,\cdots,w_k$ 表示它的子节点，$n_w(b)$ 表示起始于 w 且估值不低于 b 的请求数量。则有状态转移式

$$a(w,b)=b\cdot n_w(b)+\sum_i \max_{b'\geq b} a(w_i,b')$$

也就是把 $a(w,b)$ 的贡献拆成从 w 出发的那些请求的贡献，和从子节点出发的那些请求的贡献。

关键的观察是，在无限供应的情况下，我们只需要考虑多项式数量的成本 b。我们可以证明所有路径的价格都是某个估值：如果不是估值，可以将其价格略微提高而不会造成嫉妒或是消费者减少。

因此在 $\max\limits_{b'\geq b}$ 的计算中只需要考虑 $b'\in R_{w_i}$ 的情况，因此状态数和转移复杂度都是多项式级别的。

对于有限供应的情况，可以扩展上面的方法获得一个伪多项式时间的动态规划算法。假设边容量为 c_e。令 $a(w,c,b)$ 表示在最多 c 个来自 R_w 的请求可行时获得的最大收入，且 w 到根的路径总价格恰好为 b。

使用 $n_w^+(b)$ 表示起始于节点 w 且估值严格大于 b 的请求数量。e_i 表示 w 到子节点 w_i 的边。状态转移方程为

$$a(w,c,b)=b\cdot c_0+\sum_{i\geq 1}\max_{b'\geq b} a(w_i,c_i,b')$$

其中需要满足 $0\geq c_i\geq c_{e_i}, n_w^+(b)\leq c_0\leq n_w(b), \sum\limits_i c_i\leq c$。

因为有限供应，状态数不一定是多项式级别的，该算法只是一个伪多项式时间算法。

2.5.2 公路问题

另一个收费站问题的特例是当图实际上是一条链，并且所有请求都是链上的子路径（不一定共享公共端点），我们称之为公路问题。尽管图的结构已经是最简单的了，但这个问题仍然出奇地复杂。在下一章中我们会证明这个问题是 APX-hard 的。

然而当某些参数有界时，可以导出伪多项式时间的动态规划算法。此处再次假设边容量是无限的。算法首先依赖于以下整数性引理：

引理 7. *如果所有估值 v_i 是整数，那么存在一个最优解，其中所有价格 p_e 都是整数。*

我们可以依赖上述引理来为下面的特殊情况推导出一个伪多项式时间的动态规划算法。

定理 7. *1. 如果所有估值 v_i 是整数且存在常数上限 B，那么存在一个 $O(B^{B+2}n^{B+3})$ 的动态规划算法来找到最优价格向量。*

2. 如果所有估值 v_i 是整数且存在常数上限 B，且所有请求的路径长度都被某个常数 k 限制，那么存在一个 $O(B^{k+1}n)$ 的动态规划算法来找到最优价格向量。

证明. 我们对这两种限制给出算法的概述分析。

1. 此处动态规划维护 $a_{j,(k_1,\gamma_1),\cdots,(k_{B+1},\gamma_{B+1})}$，它表示在 j 左边最近的 $B+1$ 条边价格不为 0 的情况下（也就是 $k_1 < \cdots < k_{B+1} \le j$ 这些边，要求 $\gamma_1,\cdots,\gamma_{B+1} > 0$），可以从那些右端点 $r_i \le j$ 的请求中获得的总最大利润。

   根据整数性引理，我们只需要考虑 $\gamma_b \in \{1,\cdots,B\}$ 的情况。因此 $\sum_{b=1}^{B+1} \gamma_b > B$，这意味着对于 $l_i < k_1$ 且 $r_i \ge k_{B+1}$ 的请求是不可行的，这也保证了上述状态设计的合理性，最多只需要维护 $B+1$ 条边。

   关于状态转移，对于 $a = a_{j,(k_1,\gamma_1),\cdots,(k_{B+1},\gamma_{B+1})}$，考虑边 $j+1$，如果 $j+1$ 定价为 0，它将不出现在状态中，那么 a 将用于更新 $a_{j+1,(k_1,\gamma_1),\cdots,(k_{B+1},\gamma_{B+1})}$；否则 $j+1$ 定价 γ 时，a 将用于更新 $a_{j+1,(k_2,\gamma_2),\cdots,(k_{B+1},\gamma_{B+1}),(j+1,\gamma)}$。

   可以看出来，总状态数是 $O(B^{B+1}n^{B+2})$，每个状态转移需要 $O(nB)$ 时间，因此总时间是 $O(B^{B+2}n^{B+3})$。

2. 此处动态规划维护 $a_{j,\gamma_1,\cdots,\gamma_k}$，表示边 $j, j-1,\cdots,j-k+1$ 定价为 $\gamma_1,\cdots,\gamma_k$ 时，可以从那些右端点 $r_i \le j$ 的请求中获得的总最大利润。

   状态转移式

   $$a_{j+1,\gamma_1,\gamma_2,\cdots,\gamma_k} = \max_{\gamma_0 \in \{0,\cdots,B\}} (\gamma_1 \cdot n_{j+1,\gamma_1,\cdots,\gamma_k} + a_{j,\gamma_2,\cdots,\gamma_k,\gamma_0})$$

   也就是枚举边 $j-k+1$ 的定价 γ_0，将贡献拆成在 $j+1$ 处结束的请求贡献和在之前结束的请求贡献，取最大值。

   同样地根据整数性引理，γ_0 只需要枚举 $O(B)$，总状态数又是 $O(B^k n)$，因此运行时间为 $O(B^{k+1}n)$。

□

综上，我们推导和分析出了两种特例的算法，可以发现这和背包问题的动态规划解法很相似，复杂度不仅和问题规模 n, m 有关，还都和值域有关，因此也只能是伪多项式时间的而非多项式时间。

下一章，我们将会对无限供应情况下的无嫉妒单一意图定价问题展开更深入的讨论。

第 3 章　对 Single-minded Unlimited Supply Problems 的深入分析

3.1　问题定义

上一章我们讨论了 SMEFP 问题，本章将在上一章的基础上深入讨论无限供应下的 SMEFP 问题，即单一意图无限供应定价问题（Single-minded Unlimited Supply Pricing Problems，下文用 SUSP 代替），定义与前文中相同，这里就不再赘述。不过这里主要讨论集合及其价值，而不是竞标者及其竞标。我们给出如下的两个定义：

定义 2. *SUSP 由一个商品集合 U 和一个子集集合 $\mathcal{S}$ 定义，其中每个集合 $S \in \mathcal{S}$ 有价值 $w(S)$。我们希望为商品分配价格 $c^* : U \to \mathbb{R}_0^+$，使得*

$$c^* \in argmax_c \sum_{S \in \mathcal{S}: w(S) \geq \sum_{e \in S} c(e)} \sum_{e \in S} c(e) \tag{3.1}$$

c^* 是一个价格分配，其最大化那些商品价格之和小于其集合价值的集合的价格之和。我们定义 $prof(c)$ 为价格分配 c 产生的总收益，即 $\sum_{e \in U} c(e)$。

我们可以简单理解为，竞标者对于每一个商品集合都有一个预算帽。如果商品价格之和超过了这个预算帽，那么这个集合就不会被购买。容易证明，上述定义和上一章的定义是本质相同的，其无嫉妒性也是显然的。

定义 3. *对于图上的 SUSP 问题（G-SUSP），我们给定一个图 $G = (V, E)$ 和路径集合 $\mathcal{P}$，一个可行的价格分配是一个映射 $c : E \to \mathbb{R}_0^+$。*

我们发现，如果集合是嵌套的，那么该问题是否定义在图上没有区别，因为其都可以转化为一个线性图，通过适当对商品进行排序即可完成。

3.2　Proof

在本节中，我们首先证明在子集是嵌套的情况下，G-SUSP 是 NP-hard 的，然后证明该问题在一定条件限制下依然是 APX-hard 的。

3.2.1　公路问题

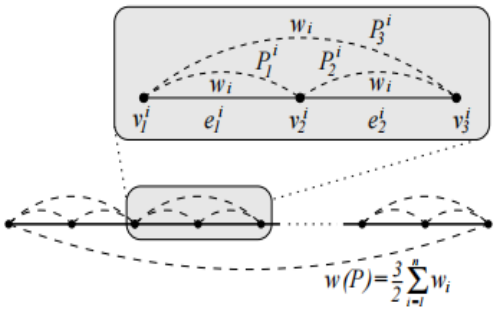
我们首先研究 G-SUSP 的一个特例，其基础的图结构只有一条线，此时可以将其看作是一个高速公路定价问题。我们假设该问题中的路径是任意长度的，对于任意两条路径 $P_1, P_2 \in P$，存在 $P_1 \subseteq P_2, P_2 \subseteq P1$ 或 $P_1 \bigcap P_2 = \emptyset$。对于所有请求集合是嵌套的 SUSP 实例都可以转化为此模型，下面是对该问题的具体分析：

首先证明该问题是 NP-hard 的。先引入 PARTITION 问题。给定一组权重 $w_1, \cdots, w_n$，找到一个子集 $S \subseteq \{1, \cdots, n\}$，使得 $\sum_{i \in S} w_i = \sum_{i \notin S} w_i$。该问题已知是 NP-hard 问题。我们可以将 PARTITION 问题多项式时间规约到 G-SUSP 问题。

对于每一个权重 w_i, 构造一个权重组件 W_i, 该权重组件由顶点 v_1^i, v_2^i, v_3^i 和边 e_1^i, e_2^i 组成。我们可以从中的得到 3 条路径 $P_1^i = \{e_1^i\}, P_2^i = \{e_2^i\}, P_3^i = \{e_1^i, e_2^i\}$，其权重为 $w(P_1^i) = w(P_2^i) = w(P_3^i) = w_i$，目标是设定价格分配 c 最大化其收益。如果 P_3^i 贡献收益，我们需要限制 $c(e_1^i) + c(e_2^i) \leq w_i$，这样的话总收益是小于等于 $2w_i$ 的，在 $c(e_1^i) + c(e_2^i) = w_i$ 时取等，为了对称我们可以假定 $c(e_1^i) = c(e_2^i) = \frac{1}{2} w_i$。也可以选择不需要 P_3^i 贡献收益，此时 $c(e_1^i) + c(e_2^i) \leq 2w_i$，我们可以发现总收益依然是小于等于 $2w_i$ 的，在 $c(e_1^i) = c(e_2^i) = w_i$ 时取等。

因此想要使得收益最大化，$c(e_1^i) = c(e_2^i) = \frac{1}{2} w_i$ 或 $c(e_1^i) = c(e_2^i) = w_i$，此时 $c(W_i) = c(e_1^i) + c(e_2^i) = 2w_i$ 或 w_i。然后定义一个从 v_1^1 到 v_3^n 的路径 P，设 $w(P) = \frac{3}{2} \sum_{i=1}^n w_i$，我们可以发现，只有 $c(W_i) = 2w_i$ 和 $c(W_i) = w_i$ 的 w_i 各占 w_i 总和一半时，即 $\sum_{i \in S} w_i = \sum_{i \notin S} w_i, S = \{i | c(W_i) = 2w_i\}$，$c(e_1^1) + c(e_2^1) + \cdots + c(e_1^n) + c(e_2^n) \leq \frac{3}{2} \sum_{i=1}^n w_i$，$P$ 才可以贡献收益，此时总收益最大为 $\frac{7}{2} \sum_{i=1}^n w_i$。

这样我们就实现了将 PARTITION 问题多项式时间规约到 G-SUSP，说明该问题是 NP-hard 的。



定理 8. *嵌套路径的 G-SUSP 问题是 NP-hard 的。*

为了找到最优价格分配，我们首先提出一种伪多项式时间算法。对于一个边集 E，边数为 m，有嵌套路径集 P，路径数为 n 的 G-SUSP 例子，定义区间集 $\mathcal{I}$ 如下：

对于每一条路径 P，定义一个区间 $I = P$，然后添加一个区间包含所有的边。定义 J 为所有的区间 I 中的一个最大子区间。如果区间 $I \backslash J$ 没有被包含在 $\mathcal{I}$ 中，将其也添加进 $\mathcal{I}$ 中。

如果一个区间 I 是由路径 P 定义的，设 $w(I) = w(P)$ 为区间的权重，否则就设 $w(I) = 0$。对于任意的区间 I，设 A_b^I 表示在价格分配的边和正好为 b 的条件下，完全包含在 I 中的路径可以得到的最大收益，即 $\sum_{e \in I} c(e) = b$。这里我们可以使用动态规

划，对于区间 I，其最大子区间为 J，定义 $K = I \backslash J$，我们有：

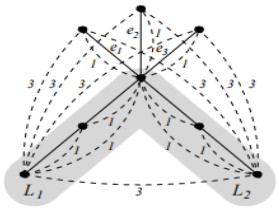
$$A_b^I = \begin{cases} \max_{b'}\{A_{b'}^J + A_{b-b'}^K\} + b & if\ b \le w(I) \\ \max_{b'}\{A_{b'}^J + A_{b-b'}^K\} & else \end{cases} \tag{3.2}$$

最后计算 $max_b A_b^E$ 即可找到最优价格分配。时间复杂度为 $O(n^3W^2)$，其中 $W = max_{p\in\mathcal{P}} w(P)$。对于任何带嵌套路径和整数估值的 G—SUSP 问题，都可以以该时间复杂度找到最优解。

引理 8. *上述算法时间复杂度为 $O(n^3W^2)$，其中 $W = max_{p\in\mathcal{P}} w(P)$。对于任何带嵌套路径和整数估值的 G—SUSP 问题，都可以以该时间复杂度找到最优解。*

3.2.2 树上收费站

对于一般的 G-SUSP 问题，上一篇文章中称其为收费站问题。已知 MAX-2SAT(3) 问题是 APX-hard 的，我们可以将 MAX-2SAT(3) 问题多项式时间规约到 G-SUSP 问题来证明 G-SUSP 问题也是 APX-hard 的。



引理 9. *设 C 是一个具有文字组件 $\mathcal{L}_1, \mathcal{L}_2$ 的子句组件，从 $\mathcal{C}$ 中可以获得的最大利润为 25. 当且仅当价格分配 c 满足 $c(\mathcal{L}_i) = 2\ c(\mathcal{L}_j) = 1, i, j = 1, 2,\ c(\mathcal{L}_1) = c(\mathcal{L}_2) = 2$ 时收益为 25，$c(\mathcal{L}_1) = c(\mathcal{L}_2) = 1$ 时收益为 24。*

这与我们在上个问题中的思想基本一致，我们希望将其归约到 G-SUSP 问题。对于每个子句 $c_i = (x_j \vee x_k)$，我们可以构造一个子句组件 C_i 在 $\mathcal{L}_h(x_j)$ 和 $\mathcal{L}_l(x_k)$ 上，$h, l \in \{0, 1, 2\}$，对于没有的组件我们用虚拟组件补充，使得对于每个变量 x_i 都有 6 个组件 $\mathcal{L}_0(x_i), \mathcal{L}_1(x_i), \mathcal{L}_2(x_i), \mathcal{L}_0(\overline{x_i}), \mathcal{L}_1(\overline{x_i}), \mathcal{L}_2(\overline{x_i})$。将一个文字组件 $\mathcal{L}_h(x_k)$ 的末端顶点和包含文字组件 $\mathcal{L}_h(\overline{x_k})$ 的子句组件 C_l 的中心顶点循环相连，由此我们可以得到一个连通图。此外，对于每一个 x_i 我们可以定义 6 条文字排除路径 $P_1^i, \cdots, P_6^i$，每一条路径都包含连接两个文字组件的 4 条边，且 $w(P) = 3$。

如果对所有文字组件 $\mathcal{L}$ 的价格分配 c 满足 $c(\mathcal{L}) \in \{1, 2\}, c(\mathcal{L}_0(x_i)) = c(\mathcal{L}_1(x_i)) = c(\mathcal{L}_2(x_i)), c(\mathcal{L}_0(\overline{x_i})) = c(\mathcal{L}_1(\overline{x_i})) = c(\mathcal{L}_2(\overline{x_i})), c(\mathcal{L}_0(\overline{x_i})) \ne c(\mathcal{L}_0(x_i))$，我们称其是 SAT-feasible 的。我们用如下方法将任意一个价格分配 c 转化为一个 SAT-feasible 的价格分配 c^*:

1. 定义 c',，如果 $c(\mathcal{L}) < 1$，则 $c'(\mathcal{L}) = 1$，如果 $c(\mathcal{L}) > 2$，则 $c'(\mathcal{L}) = 2$，否则 $c'(\mathcal{L}) = c(\mathcal{L})$。

2. 定义 $c'' = c'$，遍历所有的 P。如果 $c''(\mathcal{L}_1) \leq c''(\mathcal{L}_2), c''(\mathcal{L}_1) + c''(\mathcal{L}_2) > 3$，则 $c''(\mathcal{L}_1) = 1$。

3. 定义 c^* 是一个 SAT-feasible 的价格分配，使得 $|\{\mathcal{L}|c''(\mathcal{L}) > 1.75 \wedge c^*(\mathcal{L}) = 1\}|$ 最小。

引理 10. *对于上述的 c 和 c^*，$prof(c) \leq prof(c^*)$。*

第一步的目的是将所有的价格限制在 1 和 2 之间,这一步同时保证利润不会降低。由第一步的限制我们可以保证第二步中得到的 $c''(\mathcal{L}_1)+c''(\mathcal{L}_2) < 3$。如果 $c'(\mathcal{L}_1)+c'(\mathcal{L}_2) > 3$, 那么此时 P 利润从 0 增加，因此该步同样保证利润不会降低。

对于第三步，通过局部构建 $\mathcal{X}_\rangle = \{L_0(x_i), L_1(x_i), L_2(x_i)\}, \overline{\mathcal{X}_i} = L_0(\overline{x_i}), L_1(\overline{x_i}), L_2(\overline{x_i})$ 我们可以发现，对于集合 $B = \{\mathcal{L}|\mathcal{L} \in (\mathcal{X}_i \cup \overline{\mathcal{X}_i}) \wedge c''(\mathcal{L}) > 1.75\}$，在 SAT-feasible 的 c^* 价格分配下，不可能同时包含通过 P 连接的两个文字组件，由此可以很容易的证明该方法下利润也不会减少。

综上，我们就可以证明该引理的正确性。

对于一个给定的 MAX-2SAT(3) 的例子 I_{SAT}，其中包含 m 子句和 n 个变量，其最优真值赋值可以转化为 G-SUSP 中的最优解，利润为 $18n + 24m + 2d + 43v + opt(I_{SAT})$，其中 $opt(I_{SAT})$ 是原始 SAT 公式中满足的最大子句数，v 是由于长度为 1 的子句而增加的变量数，d 表示虚拟文字组件数量。通过放缩我们发现该式子上界为 $133 \cdot opt(I_{SAT})$。由此，我们可以得到一个 $1 - \frac{\epsilon}{134}$ 近似算法将 G-SUSP 转换为 SAT-feasible 的一个解，得到 MAX-2SAT(3) 的相应的 $(1 - \epsilon)$ 近似。我们可以得出以下结论：

定理 9. *即使限制 $l \geq 4$, 所有路径估值都是常数大小，只有三个不同的价格，每个可能的路径至多只有一个报价，每条边包含的路径数 $B \geq 8$, 图的最大度 $c \geq 7$，G-SUSP 依然是 APX-hard 的。*

3.3 Algorithm

在本节中，我们首先对于一般 SUSP 问题提出一个 $O(logB + logl)$ 的近似算法，然后使用一种新的上界方法，给出一种 $O(l^2)$ 的近似算法。

3.3.1 An $O(logl + logB)$-approximation

对于一般的 SUSP 问题，首先我们假设每个集合大小都不超过 l，并且没有任何商品出现在超过 B 个集合中。我们可以使用如下的方法来找到一个近似解：

Algorithm 5: 分区近似算法

1. 将所有的 $\delta(S)$ 四舍五入到最近的 2^k，其中 $2^k = \max \delta(S), k \in \mathbb{Z}$。然后将 T$\mathcal{S}$ 分区为 $\mathcal{S}_0, \cdots, \mathcal{S}_t$，其中 $\mathcal{S}_i = \{S \in \mathcal{S} | \delta(S) \in \{2^{i+j \cdot \lceil \log(2l^2 B) \rceil}\}\}, t = \lceil \log(2l^2 B) \rceil - 1$。

2. 对所有的 i，令 $\mathcal{S}_i^* = \mathcal{S}_i$。如果集合 $S\ T \in \mathcal{S}_i^*, \delta(S) < \delta(T)$，则将 S 从 $\mathcal{S}_i^*$ 中移除。定义价格 c_i 为：如果商品 e 在一个 $S \in \mathcal{S}_i^*$ 中的集合中，$c_i(e) = \delta(S)$，否则 $c_i(e) = 0$。

3. 返回 $c = argmax\{prof(c_i) | c_i, i = 0, \cdots, t\}$。

该算法的关键步骤是第二步，我们需要保证在删除一个集合后不会损失太多的潜在利润。我们可以证明以下引理：

引理 11. *设 $\mathcal{S}_i^*, \mathcal{S}_i$ 如上定义，那么 $\sum_{S \in \mathcal{S}_i^*} |S| \delta(S) \geq \sum_{S \in \mathcal{S}_i \backslash \mathcal{S}_i^*} |S| \delta(S)$。*

证明. 我们可以使用构建集合树的方式证明。我们从每个集合 S 的一个顶点 $v(S)$ 开始。如果 S_1 由于和 S_2 相交而从 $\mathcal{S}_i^*$ 中移除，我们将 $v(S_1)$ 作为 $v(S_2)$ 的子节点。我们可以发现，对于每一个 $v(S)$，其子节点的权重之和不超过 $v(S)$ 的权重，因此不会产生循环。在生成的树集合中，S_i^* 对应所有根节点的集合，$S_i \backslash S_i^*$ 对应所有内部节点的集合。我们可以为每一个顶点 $v(S)$ 定义一个标签 $\alpha(v(S)) = |S| \delta(S)$。

我们对其中的任意一棵树进行分析。假设该树根为 r，子节点集合为 W，对于一个顶点 v 对应于集合 S，其子节点为 $u_1, \cdots, u_j$，我们有 $\alpha(v) = |S| \delta(S) \geq \delta(S)$。由于 S 中最多包含 l 个商品，并且这些商品每一个最多都只能被包含在其他 $B-1$ 的集合中，因此 v 的子结点个数 $j \leq l \cdot B$。由算法的第一步，我们可以知道，对于包含顶点 u_i 的集合 S_i，$\delta(S_i) \leq (2l^2 B)^{-1} \delta(S) \Rightarrow |S_i| \delta(S_i) \leq (2lB)^{-1} \delta(S)$。综上，我们有：

$$\sum_{i=1}^{j} \alpha(u_i) \leq lB \cdot (2lB)^{-1} \delta(S) \leq \frac{1}{2} \alpha(v) \tag{3.3}$$

再令 $L(j)$ 表示树中所有高度为 j 的节点的集合，通过归纳我们可以得到：$\sum_{v \in L(j)} \alpha(v) \leq 2^{-j} \alpha(r)$，再对其求和即可得到 $\alpha(r) \geq \sum_{v \in W} \alpha(v)$，对每一棵树都成立，这也就证明了该引理。 □

定理 10. *上述算法得到了 $SUSP$ 的 $O(logl + logB)$-近似解。*

证明. 令 $\delta(S)$ 表示算法第一步后得到的舍入值，我们可以发现：对于任意的 $S, T \in \mathcal{S}_i^*, S \cap T = \emptyset$，我们有 $\delta(S) = \delta(T)$。因此，我们可以得到 $prof(c_i) = \sum_{S \in \mathcal{S}_i^*} |S| \delta(S), i = 0, \cdots, t$，而 $c = argmax\{prof(c_i) | c_i, i = 0, \cdots, t\}$，因此我们可以有：

$$\begin{aligned} prof(c) &\geq \frac{1}{t+1} \sum_{i=0} t \sum_{S \in \mathcal{S}_i^*} |S| \delta(S) \\ &\geq \frac{1}{2(t+1)} \sum_{i=0} t \sum_{S \in \mathcal{S}_i} |S| \delta(S) \\ &= \frac{1}{2(t+1)} \sum_{S \in \mathcal{S}} |S| \delta(S) \geq \frac{1}{4(t+1)} opt \end{aligned} \tag{3.4}$$

由于 $t = O(\log(lB))$，结论得证。 □

3.3.2 An $O(l^2)$-approximation

本节中我们首先考虑一种 SUSP 的特殊情况，即集合中的商品价格都是相等的，以下是其形式化定义：

定义 4. 平滑单一意图无限供给定价问题（*smoothed Single-minded Unlimited Supply Pricing Problems*，下文用 *s-SUSP* 代替）输入与一般的 *SUSP* 相同，我们希望找到价格分配 c^*，使得 $\sum_{S\in\Lambda(c)}\sum_{e\in S} c(e)$ 最大化，其中 $\Lambda(c) = \{S \in \mathcal{S} | c(e) \le w(S)/|S|, \forall e \in S\}$。

对于一个 SUSP 的例子 $\mathcal{I}$, 通过比较在 SUSP 和 s-SUSP 中的最优利润 $opt(\mathcal{I})$ 和 $opt_s(\mathcal{I})$，我们可以发现 $opt_s(\mathcal{I}) \ge \frac{1}{l} opt(\mathcal{I})$。根据定义我们也很容易就可以想到 $prof(c) \ge prof_s(c)$，因此想要得到 SUSP 的一个 $O(l^2)$-近似解，我们只需要找到 s-SUSP 的一个 $O(l)$-近似解即可。

首先考虑一个商品 e，对于一个给定的价格分配 c，我们定义冲突集合 $C_c(e) = \{S | \exists e' : e, e' \in S \wedge c(e) \le w(S)/|S| < c(e')\}$ 表示该集合中集合价格与 e 不冲突，但与该集合中其他某个商品价格冲突。我们同样定义非冲突集合 $N_c(e) = \{S | e \in S \wedge c(e') \le w(S)/|S|, \forall e' \in S\}$，令 $C_c = \bigcup_{e\in U} C_c(e)$。

定义 5. 设 c 是任意一个价格分配，与之对应的冲突图是一个带标签的有向多超图 $H = (V_H, E_H)$，其中每个商品 $e \in U$ 有一个顶点 v_e，每个集合 $S \in C_c$ 有一个有向超边 $f_S = (P, Q)$，其中 $P = \{v_e | S \in C_c(e)\}, Q = S \backslash P$。对于每个超边 f_S，若 $v_e \in P, f_S \in Out(v_e)$，否则 $f_S \in In(v_e)$。我们定义标签 $\alpha(v_e) = c(e) \cdot |\mathcal{N}_c(e)|$。对于所有的 $v_e \in V_H$，我们定义 $\alpha(v_e, f_S) = c(e)$ 使得 $f_S \in Out(v_e)$，并且对于所有的 $f_S = (P, Q) \in E_H$，定义 $\alpha(f_S) = \sum_{v_e\in P} \alpha(v_e, f_S)$ 使得 $f_S \in In(v_e)$。最后，对于所有的 $v_e \in V_H$，定义 $c(v_e) = c(e)$。

根据定义，超图中的每一个的顶点就对应着 s-SUSP 例子中的一个商品，每一条超边代表一个冲突集合，顶点的标签 $\alpha(v_e)$ 就是在价格分配 c 下由非冲突集合带来的对应商品 e 的利润。而超边的标签 $\alpha(f_S)$ 就是在没有冲突的情况下集合 S 的利润。标签 $\alpha(v_e, f_S)$ 就是由于 S 冲突商品 e 损失的利润。为方便下文中我们令 $H = (V_H, E_H), V' \subseteq V_H, E' \subseteq E_H$，定义 $\alpha(V') = \sum_{v\in V'} \alpha(v), \alpha(E') = \sum_{f\in E'} \alpha(f), \alpha(H) = \alpha(V_H) + \alpha(E_H)$。我们通过如下方法进行变换：

Algorithm 6

1. 对所有的 $e \in E$，假设其他商品价格均为 0，找到 s-SUSP 目标下的最优价格 $c^*(e)$。

2. 为价格分配 c^* 构建冲突图 $H_1 = (V_1, E_1)$。

3. 令 $V_1 = \{v1, \cdots, v_m\}$，其中 $c(v_1) \leq \cdots \leq c(v_m)$，对于每一个 i，检查是否满足 $\sum_{f \in In(v_i)} \alpha(f) > \frac{1}{2} \sum_{f \in Out(v_i)} \alpha(v_i, f)$，如果满足，将每个 $f = (P, Q) \in Out(v_i)$ 替换为 $f' = (P \backslash \{v_i\}, Q)$，并更新 $\alpha(f')$，生成的图为 $H_2 = (V_2, E_2)$，其中 $V_2 = V_1$。

4. 令 $R = \{v \in V_2 | Out(v) = \emptyset\}, \mathcal{E} = \{(P, Q) \in E_2 | Q \subseteq R\}$。如果 $\alpha(\mathcal{E}) \geq \alpha(R)$，那么对于所有的 $(P, Q) \in \mathcal{E}, v \in P$，设 $\alpha(v) = \alpha(v) + \sum_{f \in Out(v) \cap \mathcal{E}} \alpha(v, f)$，对于所有的 $v \in R$，设 $c(v) = \alpha(v) = 0$。最后删除 H_2 中的所有边，将生成的图即为 $H_3 = (V_3, E_3) = (v_2, \emptyset)$。

5. 对于与 H_3 中的顶点 v_e 对应的商品 $e \in U$，设 $c(e) = c(v_e)$。

该算法在 1，2 两步构建出 c^* 的冲突图后，第 3 步我们对冲突图进行变换使其满足某些特性，第 4 步使得我们最后得到的冲突图依然是一个满足 s-SUSP 的图。下面是对算法的详细证明：

引理 12. *设 $(U, \mathcal{S}, w)$ 是原始 s-SUSP 实例，则图 H_3 是价格分配 c 在 s-SUSP 实例 $(U, \mathcal{S}', w)$ 上定义的冲突图，其中 $\mathcal{S}' \subseteq \mathcal{S}$。*

证明. 如果第 4 步中 $\alpha(\mathcal{E}) \leq \alpha(R)$，那么我们只需要将 H_2 中所有超边删除，我们可以设 $\mathcal{S}' = \mathcal{S} \backslash \mathcal{C}$，其中 $\mathcal{C}$ 是 H_2 中所有的冲突集合，结论显然成立。

否则，我们就有 $R' = \{v \in V_2 | Out(v) \cap \mathcal{E} \neq \emptyset\}$。根据定义，我们将所有 $v \in R, c(v) = 0$，因此此时这些顶点变得不再冲突，将其余的超边从冲突图中移除，我们依旧可以通过从 $\mathcal{S}$ 中移除其对应的集合来得到 $\mathcal{S}'$，结论得证。 □

由于冲突图 H_3 中不再包含超边，根据上述引理，价格分配 c 也就不会在 $\mathcal{S}'$ 中产生冲突，因此，c 在 s-SUSP 目标下实现的总利润为 $\alpha(H_3)$。我们继续比较 $\alpha(H_3)$ 和 $\alpha(H_1)$：

引理 13. *我们有 $\alpha(H_2) \geq \frac{1}{2l-1} \alpha(H_1)$。*

证明. 根据算法，设 $c(v_1) \leq \cdots \leq c(v_m)$，我们用 $In_i(v_i)$ $Out_i(v_i)$ 表示第 3 步中处理顶点 v_i 之前的传入和传出超边集合。用 $\alpha_i(f)$ 表示此时超边 f 的标签。根据冲突定义，我们有：

$$(P, Q) \in \mathcal{E}_\infty \Rightarrow c(v) < c(w), \forall v \in P, w \in Q \tag{3.5}$$

这同样说明，如果 $j > i$，那么 $In^i(v_i) \cap Out^j(v_j) = \emptyset$。因此，对于任意的 $f \in In^i(v_i), In^i(v_i) \subseteq E_2$，由于不再冲突，$In^i(v_i)$ 中的超边都保持不变，我们有 $\alpha^i(f) = \cdots = \alpha^{m+1}(f)$。我们定义 W 表示从传入超边 $Out^i(v_i)$ 中移除的顶点 v_i 的集合。我们有：

$$\begin{aligned}\sum_{f\in E_2}\alpha(f) &\geq \alpha_{m+1}(\bigcup_{v_i\in W} In^i(v_i)) \\ &\geq \frac{1}{l-1}\sum_{v_i\in W}\sum_{f\in In^i(v_i)}\alpha^{m+1}(f) \\ &= \frac{1}{l-1}\sum_{v_i\in W}\sum_{f\in In^i(v_i)}\alpha^{i}(f) \\ &> \frac{1}{2l-1}\sum_{v_i\in W}\sum_{f\in Out^i(v_i)}\alpha(v,f) \\ &= \frac{1}{2l-1}(2\alpha(E_1)-\alpha(E_2))\end{aligned} \tag{3.6}$$

其中第二个不等式是因为对于所有的 $(P,Q)\in E_1, |Q|\leq l-1$。由此我们就可以得到：

$$\alpha(E_2) > \frac{1}{2l-1}\alpha(E_1) \tag{3.7}$$

同时我们也有 $\alpha(V_2)=\alpha(V_1)$，因此 $\alpha(H_2)\geq \frac{1}{2l-1}\alpha(H_1)$，引理得证。 □

引理 14. 我们有 $\alpha(H_3)\geq \frac{1}{3}\alpha(H_2)$。

证明. 根据算法，令 $R=\{v\in V_2|Out(v)=\emptyset\}, \mathcal{E}=\{(P,Q)\in E_2|Q\subseteq R\}$，定义 $A=\{v\in V_2|In(v)=\emptyset\}, B=V_2\backslash(A\cup R)$。不失一般性，假设 $B=v_1,\cdots,v_k$，其中 $c(v_1)\leq\cdots\leq c(v_k)$。初始化 $X^0=\bigcup_{v\in A}Out(v), Y^0=\emptyset$，如果 $f\in X^0$，定义标签 $\gamma(f)=\alpha(f)$，否则 $\gamma(f)=0$。同样，我们对任意的 $F\subseteq E_2$，定义 $\gamma(F)=\sum_{f\in F}\gamma(f)$。首先我们有 $\gamma(X^0)\geq\gamma(Y^0)$。然后我们重复以下过程：

1. 令 $X^j=(X^{j-1}\backslash In(v_j))\cup Out(v_j)$
2. 令 $Y^j=Y^{j-1}\cup In(v_j)$
3. 对每一个 $f\in Out(v_j)$，令 $\gamma(f)=\gamma(f)+\alpha(v_j,f)$

接下来就可以体现出算法第 3 步变换的用处了，我们有：

$$\sum_{f\in Out(v_j)}\alpha(v_j,f)\geq 2\cdot\sum_{f\in In(v_j)}\alpha(f) \tag{3.8}$$

根据定义，我们知道 $\gamma(f)\leq\alpha(f)$ 恒成立，因此我们可以得到：

$$\gamma(X^j)-\gamma(X^{j-1})\geq\sum_{f\in Out(v_j)}\alpha(v_j,f)-\sum_{f\in In(v_j)}\alpha(f)\geq\sum_{f\in In(v_j)}\alpha(f) \tag{3.9}$$

同样的，我们也有：

$$\gamma(Y^j)-\gamma(Y^{j-1})\leq\sum_{f\in In(v_j)}\gamma(f)\leq\sum_{f\in In(v_j)}\alpha(f) \tag{3.10}$$

因此，我们就会发现 $\gamma(X^j) \geq \gamma(Y^j)$。直到最后一步，我们有 $\gamma(X^k) \cup \gamma(Y^k) = E_2, \gamma(X^k) = \alpha(X^k), \gamma(Y^k) = \alpha(Y^k), X^k = \mathcal{E}$。这即说明：

$$\alpha(\mathcal{E}) = \alpha(X^k) \geq \frac{1}{2}(\alpha(X^k) + \alpha(Y^k)) = \frac{1}{2}\alpha(E_2) \tag{3.11}$$

另外，我们知道：$\max\{\alpha(\mathcal{E}, \alpha(R))\} \geq \frac{1}{3}(2\alpha(\mathcal{E}) + \alpha(R)) \geq \frac{1}{3}(\alpha(E_2) + \alpha(R))$，由于算法要么移除 $\mathcal{E}$ 要么移除 R，因此我们有：

$$\begin{aligned} \alpha(H_3) &= \alpha(H_2) - \alpha(E_2) - \alpha(R) + \max\{\alpha(\mathcal{E}), \alpha(R)\} \\ &\geq \alpha(H_2) - \frac{2}{3}(\alpha(E_2) + \alpha(R)) \geq \frac{1}{3}\alpha(H_2) \end{aligned} \tag{3.12}$$

引理得证。 □

综上，我们可以得到以下定理：

定理 11. *上述算法得到了一个适用于大小最多为 l 的集合的 SUSP 问题的 $O(l^2)$-近似解。*

证明. 根据上述引理，我们有：

$$prof_s(c) = \alpha(H_3) \geq \frac{1}{6l-3}\alpha(H_1) \geq \frac{1}{6l-3}opt_s(\mathcal{I}) \tag{3.13}$$

这样我们就得到了一个 s-SUSP 问题的 $O(l)$-近似解，根据上文所述，我们也就得到了 SUSP 问题的 $O(l^2)$-近似解。 □

第 4 章 Unit Demand Envy-free Problems

4.1 问题定义

我们首先给出单需求无嫉妒利润最大化问题 (Unit Demand Envy-free Profit-Maximization Problems, 下文用 UDEFP 代替) 的定义。首先我们对该问题的一些基本概念进行解释：

1. 单需求消费者：消费者可能对多个商品感兴趣，但是最多只会从感兴趣的商品中购买一件。

2. 无嫉妒解决方案：同上一章，每个顾客对自己的分配感到满意。

此外，UDEFP 还涉及到商品供应量的问题，分为有限供应和无限供应两种情况。本节将会对有限供应的情况提出对数级别的近似算法，同时对一些 UDEFP 的特殊情况给出多项式时间算法或伪多项式时间算法。

问题的形式化定义与 SMEFP 问题类似，主要区别在于顾客需求的不同。此处每个顾客 i 可能对会多个商品感兴趣，消费者对感兴趣的商品估值 $v > 0$，否则 $v = 0$。需要注意的是，此处消费者的每个感兴趣的商品子集大小一定为 1。

除此之外，我们定义需求集的概念，这是在确定的价格向量 $\mathbf{p}$ 条件下得到的。价格向量确定了，就能够计算每个消费者 i 从每个商品集 S 中能获得的效用 $U_i(S) = v_i(S) - p(S)$。消费者 i 的需求集 D_i 包含所有使得他获得最大效用的商品，形式化地 $D_i = \{S | U_i(S) = \max_{S'} U_i(S')\}$。

4.2 有限供应 UDEFP 问题近似算法

在本节中，我们将设计一个多项式时间的近似算法，以解决有限供应下的 UDEFP 问题。

4.2.1 二分图匹配

在对单需求情景的进一步讨论中，将分配视为二分图中的匹配会有助于理解。

具体来说，给定一个价格向量 $\mathbf{p}$，需求图是消费者 i 和物品 j 之间的二分图，其中物品 j 被复制了 $\min(u_j, n)$ 次，仅当 j 属于 D_i 时存在边 (i, j)。无嫉妒匹配是一个匹配 M，使得每个在价格 $\mathbf{p}$ 下需求集 D_i 非空的消费者 i 都被匹配。当且仅当需求图存在一个无嫉妒匹配时，价格 p 是无嫉妒的，算法输出价格和匹配的二元集 (p, M)。

事实上，我们可以研究每个物品只有一个副本的情况，即对于所有物品 j 都有 $u_j = 1$。因为在单需求场景下，问题都可以转化成物品只有一个副本的情况而不改变本质：如果一个物品有 c_j 个副本，我们可以在输入中将其替换为 $\min(c_j, n)$ 个不同的物品，并且给予每个用户对所有这些不同物品相同的估值。

4.2.2 瓦尔拉斯均衡

定义 6. 瓦尔拉斯均衡： 给定一个估值矩阵 V，瓦尔拉斯均衡 $(\mathbf{p}, M)$ 由无嫉妒定价 $\mathbf{p}$ 和匹配 M 组成，使得所有未匹配的物品价格为零。

以下定理描述了单需求定价问题中的瓦尔拉斯均衡：

定理 12. 设 $(\mathbf{p}, M)$ 为瓦尔拉斯均衡。那么 M 是估值矩阵 V 上的最大权匹配；此外，对于任何最大匹配 M'，$(\mathbf{p}, M')$ 同样是瓦尔拉斯均衡。

上述定理说明了不同的瓦尔拉斯均衡区别主要在于价格向量 $\mathbf{p}$，而 M 一定是最大权匹配。

记 $\omega(V)$ 表示估值矩阵 V 上最大权匹配的权重，对于物品 j，记 V_{-j} 表示删除了 j 之后的估值矩阵。下面的算法能够找到具有最大物品价格的瓦尔拉斯均衡。

Algorithm 7: MaxWEQ

1. **Input**: 估值矩阵 V。
2. 对于每个物品 j，令 $p_j^* = \omega(V) - \omega(V_{-j})$。
3. **Output**: 瓦尔拉斯均衡 $(\mathbf{p}^*, M)$。

定理 13. *MaxWEQ* 能够在多项式时间内找到具有最大物品价格的瓦尔拉斯均衡 $(\mathbf{p}^*, M)$：对任意瓦尔拉斯均衡 $\mathbf{p}$，对任意物品 j 有 $p_j^* \geq p_j$。

瓦尔拉斯均衡能够提供一个有效的（无嫉妒）的价格向量，并且 MaxWEQ 算法是多项式复杂度的。然而即使是具有最高价格的瓦尔拉斯均衡，其收入也可能远非最优。在瓦尔拉斯均衡的基础上，引入保留价格是关键的一步。

4.2.3 带保留价格的瓦尔拉斯均衡

定义 7. 带保留价格的瓦尔拉斯均衡： 给定估值矩阵 V 和保留价格向量 $\mathbf{r} = (r_1, \cdots, r_m)$，具有保留价格 r 的瓦尔拉斯均衡是一个无嫉妒的定价 $\mathbf{p}$ 和分配 M，使得

1. 对所有 j，有 $p_j \geq r_j$；
2. 如果物品 j 未售出，则 $p_j = r_j$；
3. 如果物品 j 在消费者 i 的需求集中且 j 未售出，则消费者 i 一定会被分配到一个物品。

给定一个任意的计算瓦尔拉斯均衡的算法 Algo，我们可以使用该算法得到一个计算具有保留价格 $\mathbf{r}$ 的瓦尔拉斯均衡的算法 Algor。

具体而言，我们通过为每个物品 j 创建两个虚拟消费者，他们对物品 j 的估值为 r_j，对所有其他物品的估值为 0，将估值矩阵 V 扩展为新的矩阵 V'。运行 Algo(V′) 后，我们得到一个瓦尔拉斯均衡 $(\mathbf{p}, M')$。在 M' 中，移除所有虚拟消费者及它们的边；最后，当有一个未售出的物品 j 在一个未被分配物品的真实消费者 i 的需求集中时，我们将物品 j 分配给消费者 i。最终的匹配 M 及价格 $\mathbf{p}$ 是就是算法 Algor(V) 的输出。

以下证明 Algor 输出结果是一个具有保留价格 $\mathbf{r}$ 的瓦尔拉斯均衡。

证明. 首先考虑 $(\mathbf{p}, M')$。对于每个物品 j，至少有一个虚拟消费者未被分配物品 j。由于该消费者没有嫉妒，可以得到 $p_j \geq r_j$。

如果物品 j 最后在 M 中未售出，那么说明一定在 M' 中分配给了虚拟消费者，这说明 $r_j \geq p_j$。因此 $p_j = r_j$。

算法的最后一步保证了满足条件 (3)。因此 $(\mathbf{p}, M)$ 是具有保留价格 $\mathbf{r}$ 的瓦尔拉斯均衡。 □

结合上文所提到的 MaxWEQ 算法，可以得到对应的带保留价格的 MaxWEQr 算法。

4.2.4 提出近似算法

在正式提出近似算法之前，先给出以下引理。

引理 15. 设 π 为估值矩阵 V 上的最大权匹配。给定一个值 r，设 k_r 为 π 中估值至少为 r 的边的数量。如果 $(\mathbf{p}, M)$ 是带保留价格 $\mathbf{r} = (r, r, \cdots, r)$ 的任何瓦尔拉斯均衡，则 $(\mathbf{p}, M)$ 的卖家利润至少为 $\dfrac{r \cdot k_r}{2}$。

证明. 考虑 π 中估值至少为 r 的边 (i, j)，其中 i 是消费者，j 是物品。根据带保留价格的瓦尔拉斯均衡定义，如果物品 j 没有被匹配，那么 $p_j = r$。同时，如果 $j \in D_i$，那么 i 一定被匹配；如果 $j \notin D_i$，那么说明 i 的需求集的物品效用一定为正值，那就说明 i 一定会被匹配（否则违反了无嫉妒性）。

因此，对于 π 中估值至少为 r 的每条边 (i, j) 有 i 或 j 被 M 匹配。

注意到 π 本身是一个匹配，因此边与边之间不存在重复的点。因此，对于每个估值至少为 r 的边，至少会给 M 带来一个新的点。而 M 总共有 $2|M|$ 个点，因此我们有 $k_r \leq 2|M|$。

即，匹配 M 以至少 r 的价格向至少 $\dfrac{k_r}{2}$ 个消费者出售物品，带来利润至少为 $\dfrac{r \cdot k_r}{2}$。 □

下面我们正式提出对数级别的有限供应 UDEFP 问题近似算法。

Algorithm 8: 无嫉妒定价近似算法

1. **Input**: 估值矩阵 V。

2. 设 π 为 V 上的最大权匹配，$r_1 \geq r_2 \geq \cdots \geq r_l$ 为 V 上的最大权匹配中的边的估值。

3. 对每个 j，使用保留价格 $\mathbf{r} = (r_j, r_j, \cdots, r_j)$ 在 V 上运行 MaxWEQr 算法，得到 $(\mathbf{p_j}, M_j)$。

4. **Output**: 找到具有最大卖家利润的 $(\mathbf{p_j^*}, M_j^*)$。

定理 14. *无嫉妒定价近似算法在多项式时间内输出一个无嫉妒定价和匹配，其卖家利润至少为 $\frac{OPT}{2\ln n}$，其中 OPT 是最优无嫉妒卖家利润。*

证明. 首先由上面的结论可以知道输出是无嫉妒的，且运行时间显然是多项式的。

设 P 为近似算法的利润。对于所有 j 满足 $P \geq \frac{j \cdot r_j}{2}$，故 $r_j \leq \frac{2P}{j}$。

另一方面，最优无嫉妒卖家利润不会超过最大权重匹配的结果，即

$$OPT \leq \sum_j r_j \leq \sum_j \frac{2P}{j} \leq 2P \ln n$$

因此有 $P \geq \frac{OPT}{2\ln n}$。 □

至此，我们得到了一个有限供应 UDEFP 问题多项式时间近似算法的设计与分析。对于无限供应的情况，论文没有提及具体算法，但容易知道无限供应的条件比有限供应更松，应当有不弱于对数级别的近似算法。

4.3 随时间定价问题

本节将会讨论 UDEFP 问题的一个特殊情况，即随时间定价问题。在这里每个消费者 i 在时间区间 $[s_i, t_i]$ 内对该物品的估值为一个常数 v_i，而在其他时间的估值为 0。无嫉妒条件意味着消费者 i 会以低于 v_i 的最低价格在 $[s_i, t_i]$ 区间内购买该物品。可以理解为，每个时刻都有物品供购买。消费者会对某个时间段内的物品感兴趣。

定理 15. *无限供应随时间定价问题可以在多项式时间内解决。*

证明. 我们给出一个动态规划算法。对于时间点 $s < t$ 和价格 p，定义 $a_p(s,t)$ 在时间区间 (s,t) 内最低价格至少为 p 时，可以从消费者处获得的最大利润。其中这些消费者 i 满足 $s < s_i$ 且 $t_i < t$。我们要计算的是 $a_0(0,\infty)$。

如果在区间 (s,t) 内没有消费者愿意支付至少为 p 的价格，利润为 0。

否则，用 $n_{t'}^q$ 表示在 $s < s_i$ 且 $t_i < t$ 的消费者中，在 t' 时刻愿意以价格 q 购买物品的消费者数量（即 $s_i \leq t' \leq t_i$ 且 $v_i \geq q$）。

那么有转移方程

$$a_p(s,t) = \max_{q \geq p, t' \in (s,t)} (a_q(s,t') + a_q(t',t) + q \cdot n_{t'}^q)$$

也就是把 (s,t) 拆分成 (s,t'),t' 时刻和 (t',t) 互不影响的三个部分，容易知道这是满足最优子结构的。

一个重要的观察是，物品只需要在 s_i 或 t_i 的时间出售（即消费者区间的端点）而不影响结果。因为如果销售时间不在端点，可以将其稍微向左或向右移动，并不改变利润。同样地，所有的销售价格都是估值 v_i：如果物品以不等于某个估值的价格 p 出售，那么就可以略微提高 p 而不会产生嫉妒或失去消费者，从而获得更高的利润。

因此动态规划状态数是有限的，只有 $O(n^3)$ 个 $a_p(s,t)$ 状态，每个状态可以在 $O(n^2)$ 时间内完成状态转移，因此是多项式时间的算法。 □

对于有限供应的情况，事实上可以根据上面的算法进行一定修改得到一个动态规划算法。假设 t 时刻有供应限制 c_t，对于给定的 s,t，记 $n^+_{t',q}$ 表示满足 $s_i \le t' \le t_i$ 且 $v_i > q$ 的消费者数量。那么有如下转移方程：

$$a_p(s,t) = \max_{q \ge p, t' \in (s,t)} \text{of}$$

$$\begin{cases} (a_q(s,t') + a_q(t',t) + q \cdot n_{t',q}) & \text{if } n_{t',q} \le c_{t'} \\ (a_q(s,t') + a_q(t',t) + q \cdot c_{t'}) & \text{if } n^+_{t',q} \le c_{t'} \le n_{t',q} \\ -\infty & \text{if } c_{t'} < n^+_{t',q} \end{cases}$$

遗憾的是，因为容量有限，这个问题的状态数并不一定是多项式的，不能简单地像无限供应的情况一样无脑转移，因此上述动态规划实际上是一个伪多项式算法。

事实上，有限供应下的随时间定价问题能否在多项式时间解决仍是 Open Problem，也并不清楚该问题是否是 NP-Hard，目前已知的最好的近似算法就是上一节分析的对数近似算法。

4.4 诚实竞争机制

研究无嫉妒定价的另一个动机是，由此定价所能获得的利润是用于分析最大化利润的组合拍卖诚实机制的一个自然下界。在组合拍卖中，估值只有消费者自己知道，卖方并不知道。卖方需要在进行拍卖的过程中，以隐含或显式的方式向消费者征求估值。

了解拍卖机制的消费者如果通过谎报其估值能从结果中获得更多效用，就会选择这样做。一个主流的解决方案是设计一个诚实机制，确保消费者报告其真实估值是最优的策略。

本节将会提出一个诚实机制，对于所有估值在 $[1,h]$ 区间内的单需求拍卖，能够保证 $O(\log h)$ 的近似比。

4.4.1 VCG 机制

VCG 是一个很经典的诚实机制，它与瓦尔拉斯均衡非常相似，输出一个由最大权匹配 M 给出的高效分配。不同的是，VCG 旨在实现报价真实性，因此计算出的价格不同于上面的 MaxWEQ 算法。具体而言，M 分配给消费者 i 的物品 j 时，i 需要支付的价格是

$$p_i(j) = v_i(j) - \omega(V) + \omega(V_{-i})$$

其中 $\omega(V)$ 仍然是表示估值矩阵 V 最大匹配的权重。

VCG 不仅是已知的诚实机制，更有趣的是它的价格正好是最低的瓦尔拉斯价格。因此，VCG 机制的价格是无嫉妒的。因为 VCG 机制计算了瓦尔拉斯均衡，那么就可以使用 3.2.3 节介绍的算法来获得带保留价格的 VCG 机制即 VCGr。并且，已有工作证明了这里生成的 VCGr 机制是诚实的。

4.4.2 利用 VCGr 机制

VCGr 机制是诚实的，我们可以利用这一事实，以一种类似于有限供应 UDEFP 近似算法的方式获得一个诚实的、$\log h$ 级别近似的竞争拍卖机制。然而，需要强调的是，为了保证真实性，我们并不能为不同的保留价格运行 VCGr 算法并取最优，因为这会导致了解拍卖机制的消费者不会选择诚实报告估值而是谋求更多效用；相反地，我们在若干个候选保留价格中随机选择保留价格运行 VCGr。这样可以避免确定性算法带来的不诚实性。完整机制如下：

Algorithm 9: 单需求组合拍卖

1. **Input**: 估值矩阵 V，其中对于任意 i, j，估值满足 $1 \le v_i(j) \le h$。
2. 从 $\{1, \cdots, \log h\}$ 中均匀随机选择一个整数 k，令 $r = 2^k$。
3. 以保留价格 $\mathbf{r} = (r, r, \cdots, r)$ 在估值矩阵 V 上运行 VCGr 算法，得到 $(\mathbf{p}, M)$。
4. **Output**: $(\mathbf{p}, M)$。

定理 16. *对于所有消费者估值在 $[1, h]$ 内的任意输入，上述单需求组合拍卖机制是诚实且 $\log h$ 级别近似的竞争拍卖机制。*

证明. 拍卖的诚实性直接来自 VCGr 算法的诚实性。

设 $r_1 \ge r_2 \ge \cdots \ge r_m$ 为在最大权匹配 M 中售出的物品价格。显然最优利润的上界为 $\sum_j r_j$。设 n_r 为在 M 中以价格 r 或更高价格售出的物品数量（即 $r_j \ge r$ 的 j 的数量）。

保留价格 $\mathbf{r} = (r, r, \cdots, r)$，VCGr 的收入至少为 $\frac{r \cdot n_r}{2}$。因此拍卖的期望收入至少为

$$R \geq \sum_{k=0}^{\lfloor \log h \rfloor} \frac{2^k \cdot n_{2^k}}{2 \log h}$$

另一方面，对于一个价格 r_j，根据其二进制表示可以将其限制为

$$r_j \leq 2 \sum_{0}^{\lfloor \log h \rfloor} 2^k \cdot \left[r_j \geq 2^k \right]$$

其中 $\left[r_j \geq 2^k \right]$ 表示，如果 $r_j \geq 2^k$，则为 1，否则为 0。

对所有 j 求和有 $\sum_j r_j \geq 2 \sum_{k=0}^{\lfloor \log h \rfloor} 2^k \cdot n_{2^k}$

上面的式子相当于对二进制每一位求和，$r_1, r_2, \cdots, r_m$ 共有 n_{2^k} 个比 2^k 大的数。

而

$$2 \sum_{k=0}^{\lfloor \log h \rfloor} 2^k \cdot n_{2^k} \leq (4 \log h) \cdot R$$

完成证明。 □

至此，我们得到了一个诚实的、$\log h$ 近似的竞争拍卖机制。

参考文献

[1] V.Guruswami, J.D.Hartline, A.R.Karlin, D.Kempe, C.Kenyon, and F.McSherry. On profit-maximizing envy-free pricing. In Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms, pages 1164-1173.

[2] P.Briest and P.Krysta. Single-minded unlimited supply pricing on sparse instances. In Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm, pages 1093-1102. Society for Industrial and Applied Mathematics, 2006.

[3] M.Cheung and C.Swamy. Approximation algorithms for single-minded envy-free profit-maximization problems with limited supply. In Foundations of Computer Science, 2008. FOCS'08. IEEE 49th Annual IEEE Symposium on,pages 35-44. IEEE, 2008.

[4] E. H. Clarke. Multipart pricing of public goods. Public Choice, 11:17-33, 1971.

[5] W. Vickrey. Counterspeculation, auctions, and competitive sealed tenders. J. of Finance, 16:8-37, 1961.

[6] T. Groves. Incentives in teams. Econometrica, 41:617-631, 1973.