充电桩覆盖收益最大化问题

—— 带容量限制的最大收益集合覆盖问题

小组成员：

曹始文 12532598

杨馥羽 12532563

CATALOGUE

SUSTech Southern University of Science and Technology

01

问题背景

问题背景

- 某城市有若干住宅楼。每栋楼都有一定数量的用户需要使用电动车充电站。每栋楼的每个用户需求被满足时，会产生一定的收益，该收益因楼而异。
- 城市规划部门已划定若干潜在区域用于安装充电站。每个区域可服务于特定的楼宇，不同区域的覆盖范围可能存在重叠。
- 如果选定某个区域进行建设，则需承担一定的固定成本。每个区域可容纳的充电站数量有限，且受物理条件、规划或其他因素的限制。

- 我们的目标是在确保每栋楼的用户都能被分配到合适区域的前提下，确定在哪些区域建设充电站以及每栋楼该如何分配充电站容量。最终目标是在预算和容量限制范围内，最大化净利润，即充电站覆盖用户的总收入减去建设成本。

02

数学建模

数学建模

参数定义

- 集合与索引：

$\mathcal{I} = \{1, 2, \ldots, n\}$：楼栋索引集合，$n$ 为楼栋总数

$\mathcal{J} = \{1, 2, \ldots, m\}$：区域索引集合，$m$ 为候选区域总数

- 输入参数：

$D_i \geq 0$：楼栋 i 的潜在用户总数

$p_i \geq 0$：楼栋 i 每覆盖一个用户带来的收益

$c_j \geq 0$：区域 j 的固定建设成本

$U_j \in \mathbb{Z}_+$：区域 j 的充电桩数量上限

$a_{ij} \in \{0, 1\}$：覆盖关系矩阵，$a_{ij} = 1$ 表示区域 j 可以覆盖楼栋 i

数学建模

决策变量

1. 建设决策变量

用0-1决策变量 z_j 表示是否在区域 j 进行建设。$z_j \in \{0, 1\}$。

2. 充电桩配置变量

用变量 x_j 表示在区域 j 内设置的充电桩数量。为了反映桩数的离散性，规定 $x_j \in \mathbb{Z}_+$。

3. 用户分配变量

用变量 y_{ij} 表示楼栋 i 中被分配到区域 j 进行充电的用户人数，对所有楼栋-区域对 (i, j) 规定 $y_{ij} \geq 0$。

数学建模

约束条件

1. 覆盖与建设逻辑约束: $y_{ij} \le D_i\, a_{ij}\, z_j, \quad \forall i \in \mathcal{I}, \forall j \in \mathcal{J}.$

2. 用户需求约束: $\sum_{j \in \mathcal{J}} y_{ij} \le D_i,$

3. 区域容量约束: $\sum_{i \in \mathcal{I}} y_{ij} \le x_j.$

4. 充电桩数量联动约束: $0 \le x_j \le U_j\, z_j.$

5. 变量类型约束: $z_j \in \{0,1\}, \quad x_j \ge 0, \quad y_{ij} \ge 0, \quad \forall i \in \mathcal{I}, \forall j \in \mathcal{J}.$

数学建模

目标函数

1. 总收益计算

所有被成功覆盖的用户带来的总收益可以表示为

$$\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} p_i\, y_{ij}.$$

2. 总成本计算

为建设各个区域支付的总固定成本为

$$\sum_{j\in\mathcal{J}} c_j\, z_j.$$

3. 净收益目标函数

我们希望最大化的净收益（即总收益减去总成本）为

$$\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} p_i\, y_{ij} - \sum_{j\in\mathcal{J}} c_j\, z_j.$$

完整数学规划模型

将上述目标函数与约束条件综合起来，可以得到该充电桩选址与配置问题的形式化数学规划模型：

$$
\begin{aligned}
\max_{z,x,y}\quad & \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} p_i\, y_{ij} - \sum_{j\in\mathcal{J}} c_j\, z_j, && \\
\text{s.t.}\quad & y_{ij} \le D_i\, a_{ij}\, z_j, && \forall i\in\mathcal{I},\ \forall j\in\mathcal{J}, \\
& \sum_{j\in\mathcal{J}} y_{ij} \le D_i, && \forall i\in\mathcal{I}, \\
& \sum_{i\in\mathcal{I}} y_{ij} \le x_j, && \forall j\in\mathcal{J}, \\
& 0 \le x_j \le U_j\, z_j, && \forall j\in\mathcal{J}, \\
& z_j\in\{0,1\},\ x_j\ge 0,\ y_{ij}\ge 0, && \forall i\in\mathcal{I},\ \forall j\in\mathcal{J}.
\end{aligned}
$$

03

复杂度分析与规约证明

复杂度分析与规约证明

考虑如下判定问题：

是否存在一组决策变量 (z, x, y) 满足所有约束

$$
\begin{aligned}
\max_{z,x,y} \quad & \sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{J}} p_i\, y_{ij} - \sum_{j\in\mathcal{J}} c_j\, z_j, && \\
\text{s.t.} \quad & y_{ij} \le D_i\, a_{ij}\, z_j, && \forall i \in \mathcal{I},\ \forall j \in \mathcal{J}, \\
& \sum_{j\in\mathcal{J}} y_{ij} \le D_i, && \forall i \in \mathcal{I}, \\
& \sum_{i\in\mathcal{I}} y_{ij} \le x_j, && \forall j \in \mathcal{J}, \\
& 0 \le x_j \le U_j\, z_j, && \forall j \in \mathcal{J}, \\
& z_j \in \{0,1\},\ x_j \ge 0,\ y_{ij} \ge 0, && \forall i \in \mathcal{I},\ \forall j \in \mathcal{J}.
\end{aligned}
$$

且目标函数满足

$$
\sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{J}} p_i\, y_{ij} - \sum_{j\in\mathcal{J}} c_j\, z_j \ge K\,?
$$

复杂度分析与规约证明

给定一个候选解 (z, x, y) ，我们可以在多项式时间内验证它是否为一个证据。

- 为该候选解进行下面三项检查：
  1. 检查线性约束
  2. 检查变量类型
  3. 计算目标函数并比较阈值
- 所有检查时间复杂度不超过 O(nm)

假设所有输入参数都以二进制编码给出，则上述检验步骤的时间复杂度为输入规模的多项式。

因此，该判定问题属于NP。

复杂度分析与规约证明

为了证明该问题是 NP-hard，我们从经典的集合覆盖问题构造一个多项式时间规约。

集合覆盖判定问题定义如下:

- 给定一个有限集合（宇宙）

$$U = \{e_1, e_2, \ldots, e_n\},$$

- 一族子集

$$\mathcal{S} = \{S_1, S_2, \ldots, S_m\}, \quad S_j \subseteq U,$$

- 以及一个正整数 k,

问：是否存在一个索引集合 $J^\star \subseteq \{1, \ldots, m\}$，满足 $|J^\star| \leq k$ 且

$$\bigcup_{j \in J^\star} S_j = U\ ?$$

复杂度分析与规约证明

规约构造步骤

步骤 1：楼栋与元素对应

将每个元素 $e_i \in U$ 对应为一栋楼 i。于是楼栋集合为

$$\mathcal{I} = \{1, \ldots, n\}, \quad n = |U|.$$

对每栋楼 i 设置潜在需求

$$D_i = 1, \quad \forall i \in \mathcal{I},$$

即每个元素只需要"被服务"一次。为简化，将每个被覆盖用户的收益设置为统一常数：

$$p_i = M, \quad \forall i \in \mathcal{I},$$

其中常数 M 取为

$$M = m + 1.$$

复杂度分析与规约证明

规约构造步骤

步骤 2：区域与子集对应

将每个子集 S_j 对应为一个候选区域 j。于是区域集合为

$$\mathcal{J} = \{1, \ldots, m\}, \quad m = |\mathcal{S}|.$$

定义覆盖关系矩阵 a_{ij} 为

$$a_{ij} = \begin{cases} 1, & \text{若 } e_i \in S_j, \\ 0, & \text{否则}, \end{cases} \qquad \forall i \in \mathcal{I}, \forall j \in \mathcal{J},$$

即区域 j 正好覆盖集合 S_j 对应的那些楼栋。

复杂度分析与规约证明

规约构造步骤

步骤 3：容量与建设成本

为避免容量约束人为限制，令每个区域的最大桩数上限足够大，例如

$$U_j = n, \quad \forall j \in \mathcal{J},$$

并允许 x_j 足以服务所有其覆盖的楼栋（因为每栋最多 1 个用户）。区域建设成本设为常数

$$c_j = 1, \quad \forall j \in \mathcal{J},$$

即每选一个区域的成本为 1。在容量足够大的情况下，只要 $z_j = 1$，就完全可以为该区域能覆盖到的楼栋提供服务。

复杂度分析与规约证明

规约构造步骤

步骤 4：阈值设定

在该充电问题上，我们关注如下目标函数：

$$\mathrm{Profit}(z,x,y)=\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}p_i\,y_{ij}-\sum_{j\in\mathcal{J}}c_j\,z_j.$$

在上述构造下，由于 $D_i=1$，每个楼栋 i 要么完全被某个区域服务（贡献 M 的收益），要么完全未被服务（贡献 0）。记被成功覆盖的楼栋集合为

$$C(J)=\{i\in\mathcal{I}:\text{存在 }j\in\mathcal{J}\text{ 使 }y_{ij}=1\},$$

记被选择的区域集合为

$$J=\{j\in\mathcal{J}:z_j=1\}.$$

此时可以将目标函数重写为

$$\mathrm{Profit}(J)=M\cdot|C(J)|-|J|.$$

我们在规约中选择阈值

$$K=Mn-k.$$

复杂度分析与规约证明

充分性证明：　**如果存在集合覆盖，则存在利润至少为 K 的解**

假设在 Set Cover 实例中，存在一个索引集合 $J^\star \subseteq \{1, \dots, m\}$，满足

$$|J^\star| \le k, \quad \bigcup_{j \in J^\star} S_j = U.$$

在充电实例中构造如下解：

- 对 $j \in J^\star$，令 $z_j = 1$；对 $j \notin J^\star$，令 $z_j = 0$；
- 对每个选择的区域 $j \in J^\star$，取足够多的桩数（例如 $x_j = n$），显然满足 $x_j \le U_j z_j$；
- 由于 $J^\star$ 是集合覆盖，对每个楼栋 i，至少存在某个 $j \in J^\star$ 使 $a_{ij} = 1$。从中任取一个区域 j，令 $y_{ij} = 1$，其余 $y_{ij'} = 0$。这样对所有楼栋 i 都有 $\sum_j y_{ij} = 1 \le D_i$，并且对每个区域 j 有

$$\sum_i y_{ij} \le n = x_j,$$

所有约束均被满足。

复杂度分析与规约证明

充分性证明： **如果存在集合覆盖，则存在利润至少为 K 的解**

此时所有楼栋都被覆盖。目标函数为

$$\text{Profit}(J^\star) = Mn - |J^\star| \geq Mn - k = K.$$

因此，若 Set Cover 实例有大小不超过 **k** 的覆盖，则充电实例中存在一个可行解使目标值至少为 **K**。

复杂度分析与规约证明

必要性证明： **如果存在利润至少为 K 的解，则存在大小不超过 k 的集合覆盖**

1. 证明该解必须覆盖所有楼栋

假设在充电实例中存在一个可行解 (z,x,y)，对应的区域集合 $J=\{j\in\mathcal{J}:z_j=1\}$ 满足

$$\mathrm{Profit}(J)=M|C(J)|-|J|\geq K=Mn-k\geq Mn-m.$$

若存在某个楼栋 i 未被服务，则 $i\notin C(J)$，因此 $|C(J)|\leq n-1$。注意 $|J|\geq 0$，故有

$$\mathrm{Profit}(J)=M|C(J)|-|J|\leq M(n-1)-0=M(n-1).$$

代入 $M=m+1$，得到

$$Mn-m=(m+1)n-m=(m+1)(n-1)+(m+1-m)=M(n-1)+1.$$

$$K\geq M(n-1)+1>M(n-1).$$

矛盾。故任何满足 $\mathrm{Profit}(J)\geq K$ 的可行解必须满足 $|C(J)|=n$，即所有楼栋都被覆盖。

复杂度分析与规约证明

必要性证明：　**如果存在利润至少为 K 的解，则存在大小不超过 k 的集合覆盖**

2. 证明目标函数等价

在 $|C(J)| = n$ 的情况下，目标函数化简为

$$\mathrm{Profit}(J) = Mn - |J|.$$

条件 $\mathrm{Profit}(J) \geq K = Mn - k$ 等价于

$$Mn - |J| \geq Mn - k \quad \Longleftrightarrow \quad |J| \leq k.$$

由于 $|C(J)| = n$，说明对每个楼栋 i 都存在 $j \in J$ 使 $a_{ij} = 1$，也即

$$\{e_i : i \in \mathcal{I}\} \subseteq \bigcup_{j \in J} S_j.$$

由构造中楼栋与元素一一对应可知

$$\bigcup_{j \in J} S_j = U,$$

且 $|J| \leq k$。这说明集合 J 在原 Set Cover 实例中给出了一个大小不超过 k 的集合覆盖。

SUSTech Southern University of Science and Technology

04

算法设计

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Maecenas porttitor congue massa. Fusce posuere, magna sed pulvinar ultricies.

算法设计——问题分解

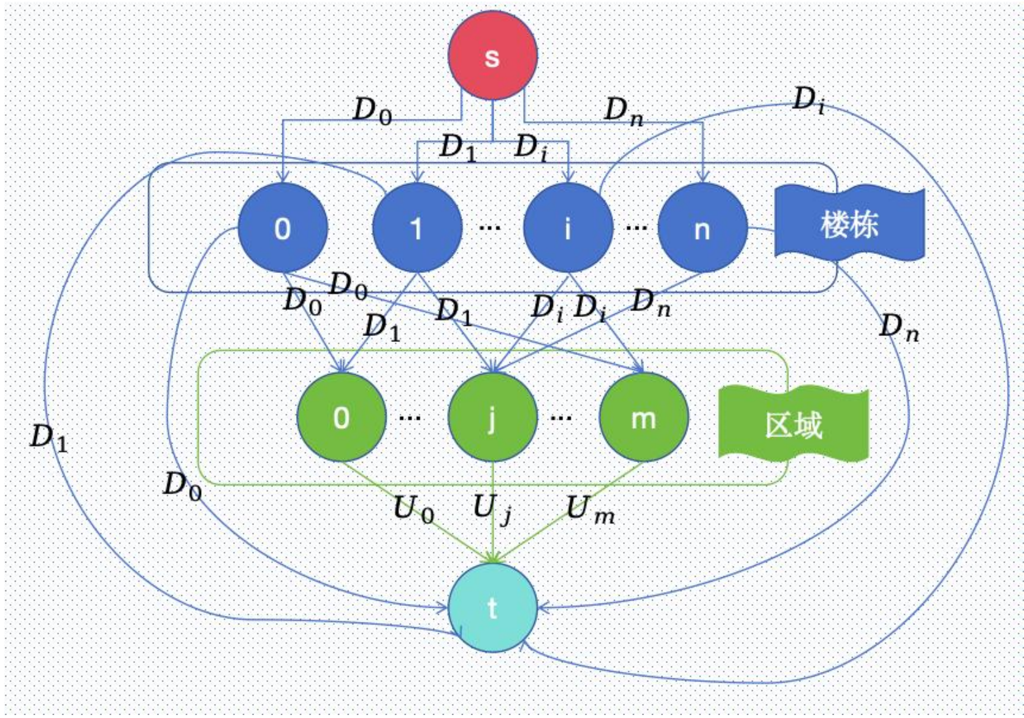
子问题：如何求解区域确定下的最优分配？

- 对于该子问题仅需考虑如何人员覆盖最大化
  - 利用**最小费用最大流算法求解**

算法设计——问题分解

子问题建模

- 节点设置
  - 源点**s**，楼栋节点集合**I**，充电区域节点集合**J**，汇点**t**



- 有向边设置
  - 源点**s** → 楼栋i：容量D_i，费用0
  - 楼栋i → 区域j： 容量D_i，费用$-p$；当且仅当区域j覆盖楼栋i时
  - 区域j → 汇点t：容量$U_j * z_j$，费用0
  - 楼栋i → 汇点t：容量D_i，费用0（表示未被服务的用户）

算法设计——问题分解

最小费用最大流

- 时间复杂度
  - 网络节点数：**O(n + m); **网络边数：**O(n*m)**
  - 总体时间复杂度：**O((n+m)^2*log(n+m)*C)**，其中**C = max{**D_i, U_j**}**是最大容量
- 空间复杂度
  - **存储网络图：**O(n*m); **存储流结果：**O(n*m)
  - **总体空间复杂度：**O(n*m)

算法设计

- 使用以下四种方法求解
  - 暴力
  - 贪心
  - 混合整数线性规划
  - 遗传算法

算法设计（一）

- 暴力求解
  - 外层枚举
    - 枚举所有 **2^m** 种建设方案
  - 复杂度分析
    - 时间复杂度: $O(2^m * (n+m)^2 * log(n+m) * C)$
    - 空间复杂度: $O(n * m)$

算法设计（二）

- 贪心算法
  - 计算区域性价比
    - $score_j = \frac{p*min\{\sum_{i;a_{ij}=1} D_i, U_j\}}{c_j}$
  - 从高到低依次尝试添加区域**j**
    - 如果可以增加收益：放入
    - 反之：跳过

算法设计（二）

- 贪心算法
  - 时间复杂度
    - 计算性价比：$O(n*m)$；排序：$O(m\, log(m))$
    - 总体时间复杂度：$O(m*(n+m)^2*log(n+m)*C)$
  - 空间复杂度
    - $O(n*m)$

算法设计（三）

- 基于**MILP**的算法
  - 决策变量：$z_j \in \{0,1\}; x_j \in Z_+;\ y_{ij} \in Z_+$
  - 约束条件：覆盖关系、需求约束、容量约束
  - 使用 **PuLP** 库调用 **CBC** 求解器（或其他 **MILP** 求解器）进行分支定界求解

算法设计（三）

- 基于**MILP**的算法
  - 时间复杂度
    - 最坏情况 $O(2^m * (n * m)^{3.5})$，实际中由于剪枝和预处理通常远小于此
  - 空间复杂度
    - 存储 **MILP** 模型：$O(n * m)$（约束矩阵）
    - 分支定界树：最坏情况 $O(2^m)$ 个节点
    - 总体空间复杂度：$O(n * m + 2^m)$，实际中由于剪枝通常远小于2^m

算法设计（四）

- 遗传算法
  - 染色体表示：二进制串 $ch \in \{0,1\}^m$，直接表示建设决策z
  - 参数设置
    - 种群规模N=50；最大代数G=100
    - 随机生成初始种群，每个个体以概率 0.3 为 1
    - 单点交叉，交叉率p=0.8
    - 变异：位翻转，变异率p=0.1
    - 精英保留：每代保留最优的 10% 个体直接进入下一代
    - 早停机制：连续20代没有提升，则提前停止

算法设计（四）

- 遗传算法
  - 时间复杂度
    - $O(G * N * (n + m)^2 * log(n + m) * C)$
  - 空间复杂度
    - $O(N * m + n * m)$

05

结果分析

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Maecenas porttitor congue massa. Fusce posuere, magna sed pulvinar ultricies.

结果分析

问题规模	方法	达到最优解次数	最优解概率	平均运行时间
n=10, m=5	贪心算法	9/10	90%	15.8ms
	MILP求解器	10/10	**100%**	**12.8ms**
	暴力枚举	10/10	**100%**	32.6ms
	遗传算法	10/10	**100%**	1.221s
n=15, m=8	贪心算法	7/10	70%	14.2ms
	MILP求解器	10/10	**100%**	**21.0ms**
	暴力枚举	10/10	**100%**	387.0ms
	遗传算法	10/10	**100%**	2.263s
n=20, m=10	贪心算法	7/10	70%	**24.0ms**
	MILP求解器	10/10	**100%**	45.9ms
	暴力枚举	10/10	**100%**	2.112s
	遗传算法	10/10	**100%**	3.452s

结果分析

SUSTech Southern University of Science and Technology

问题规模	方法	达到最优解次数	最优解概率	平均运行时间
n=25, m=12	贪心算法	7/10	70%	**37.9ms**
	MILP求解器	10/10	**100%**	45.8ms
	暴力枚举	10/10	**100%**	11.145s
	遗传算法	10/10	**100%**	5.798s
n=30, m=15	贪心算法	7/10	70%	**61.4ms**
	MILP求解器	10/10	**100%**	92.2ms
	暴力枚举	0/10	**0%**	60.0s (超时)
	遗传算法	8/10	80%	8.284s

组员分工

曹始文　12532598　分配问题分析，算法代码编写，测试样例生成，文档及ppt编写

杨馥羽　12532563　问题设计，规约分析，算法逻辑，文档及ppt编写

SUSTech Southern University of Science and Technology

PRESENTATION POWERPOINT

THANK YOU

Supporting text here. Keep your notes concise and relevant, eliminating extraneous content. Supporting text here