范数下无容量限制设施选址逆问题的求解方法

李子慷， 刘林冬， 于成成

(中国科学技术大学 管理学院国际金融研究院，安徽 合肥 230026)

摘 要: 一个优化问题的逆问题是这样一类问题，在给定该优化问题的一个可行解时，通过最小化目标函数中参数的改变量(在某个范数下)使得该可行解成为改变参数后的该优化问题的最优解。对于本是NP-难问题的无容量限制设施选址问题，证明了其逆问题仍是NP-难的。研究了使用经典的行生成算法对无容量限制设施选址的逆问题进行计算，并给出了求得逆问题上下界的启发式方法。两种方法分别基于对子问题的线性松弛求解给出上界和利用邻域搜索以及设置迭代循环次数的方式给出下界。数值结果表明线性松弛法得到的上界与最优值差距较小，但求解效率提升不大；而启发式方法得到的下界与最优值差距极小，极大地提高了求解该逆问题的效率。

关键词: 无容量限制设施选址问题；逆问题；行生成算法；启发式算法

中图分类号: O221.7 **文章标识码:** A **文章编号:** 1007-3221(2022)07-0086-07 **doi:** 10.12005/orms.2022.0220

The Method for the Inverse Uncapacitated Facility Location Problem Under One Norm

LI Zi-kang, LIU Lin-dong, YU Cheng-cheng

(*International Institute of Finance, School of Management, University of Science and Technology of China, Hefei* 230026, *China*)

Abstract: The inverse problem of a general optimization problem is to give a feasible solution by minimizing the change of parameters in the objective function under a fixed norm so that the feasible solution becomes optimal for the original optimization problem after the parameter adjustment. For the uncapacitated facility location problem-which is NP-hard, we prove that its inverse problem is also NP-hard. In this paper, the classical row generation algorithm is used to calculate the inverse problem of the uncapacitated facility location problem, and then two heuristic methods are developed to get the upper and lower bounds of the inverse problem. These two methods give the upper bound based on the linear relaxation of the subproblem and the lower bound by the local search technique, respectively. The numerical results show that the gap between the upper bound obtained by the linear relaxation method and the optimal value is small, while the efficiency is not improved significantly. However, the gap between the lower bound obtained by the heuristic method and the optimal value is very small, which greatly improves the efficiency of solving the inverse problem.

Key words: uncapacitated facility location problem; inverse problem; row generation method; heuristic algorithm

0 引言

最基本的设施选址问题分为无容量限制和有容量限制两种。其中无容量限制设施选址问题(Uncapacitated Facility Location Problem, UFLP)在文献中得到了广泛的讨论[1]。Krarup对研究UFLP问题的文献进行了综述，并指出了UFLP问题是NP-难的[2]。逆优化问题最初由Tarantola给出了逆优化问题在地球物理科学中的详细讨论[3]。Heuberger给出了关于逆优化问题较为全面的综述[4]。Ahuja和Orlin证明了在范数L_1和范数L_∞下，如果原线

收稿日期: 2020-07-19

基金项目: 国家自然科学基金优秀青年科学基金资助项目(72022018)；中国科学院青年创新促进会(2021454)；国家自然科学基金青年科学基金资助项目(71701192)

作者简介: 李子慷(1996-)，男，河南洛阳人，在读硕士，研究方向为：整数规划、逆优化；刘林冬(1990-)，通讯作者，男，江苏盐城人，特任教授，博士，研究方向为：整数规划、线性规划、合作博弈、逆优化等；于成成(1998-)，男，安徽亳州人，在读硕士，研究方向为：逆优化、离散优化。

性规划问题多项式可解则其逆问题也是多项式可解的[5]。通常来讲,只有逆问题的可行域是一个多面体,才能确保在范数 L_1下的逆问题在多项式时间内是可解的[6]。Schaefer 给出了一般整数规划逆问题的多面体描述,并提出了两种算法求解[7]。

由于设施选址逆问题在一般网络下,大多数是 NP-难的,很多学者考虑了特殊情况下的逆选址问题。Cai 等人考虑了逆单中心选址问题,证明了即使在单中心选址原问题是多项式可解的情况下,逆单中心选址问题仍然是 NP-难的[8]。Alizadeh 等人考虑了在树状图中边长度仅允许增加时的逆单中心选址问题,给出了一个时间复杂度为 $O(n\log n)$ 的精确算法[9]。Guan 给出了在加权 L_∞ 范数下,逆 1-中值选址问题线性时间算法,并证明了在加权哈明距离下该逆问题是 NP-难的[9]。

在逆选址值问题上,Berman 等人证明了在二分网络中的逆选址值问题是 NP 完全的,同时给出了在树形网络中该问题的线性表达式[10]。Zhang 等人研究树形网络中的逆选址值问题,并给出了一个时间复杂度为 $O(n^2\log n)$ 的多项式时间算法,同时提出该问题的变体,证明了逆多设施选址值问题可以转化为几个逆单设施选址值问题[11]。

设施选址逆问题具有潜在的应用前景,它为我们展示了如何花费尽可能少的成本,从而固定开启某些现有设施以提供便利。对应到模型中,即给定了开设哪些设施,这些设施对哪些顾客进行服务,目标函数则为使得设施和运输成本改变量达到最小。因而研究最基本的无容量限制设施选址逆问题及其求解算法是十分具有现实价值的。

本文研究了无容量限制设施选址逆问题(Inverse Uncapacitated Facility Location Problem, IUFLP)模型的建立与求解方法。用行生成算法将 IUFLP 问题分为主问题和子问题,其中子问题为一般的无容量限制设施选址问题(UFLP)。对子问题进行不同的处理方式,可以分别得到 IUFLP 问题的上下界。

本文安排如下: 第一节给出了 UFLP 的定义以及 IUFLP 模型的建立。第二节给出了 IUFLP 的行生成求解方法。第三节对由行生成算法得到的子问题进行了改进,得到能求出该逆问题上下界的启发式算法。第四节给出了 IUFLP 的简单示例以及计算结果。第五节给出了对于该逆问题延拓以及算法改进的展望。

1　UFLP 和 IUFLP 问题的模型

设施选址问题通常有如下的定义: 假设有 m 个设施和 n 个顾客。我们希望选择(1)哪些设施要打开,以及(2)哪些打开的设施要用于向哪些顾客提供服务,以便以最低的成本满足需求。引入以下记号,令 f_i 表示开启设施的固定成本,$i\in M$,$M=\{1,\cdots,m\}$。r_{ij}表示运输商品从设施 i 到顾客 j 的成本,$j\in N$,$N=\{1,\cdots,n\}$。无容量限制指每个设施可以提供的容量 K 是充分大的。

设为 f_i 维 m 的行向量,将 r_{ij}按行首尾相接为一个$(m\times n)$维的行向量,则可用一个$(m+m\times n)$维行向量 $c=(f,r)$ 表示成本。

则无容量限制设施选址问题(UFLP)定义如下:

$$\min\sum_{i=1}^{m}\sum_{j=1}^{n}r_{ij}u_{ij}+\sum_{i=1}^{m}f_iv_i$$

$$\text{s.t.}\ \sum_{i=1}^{m}u_{ij}=1,\forall j\in N$$

$$\sum_{j=1}^{n}u_{ij}\le Kv_i,\forall i\in N$$

$$u_{ij},v_i\in\{0,1\},\forall i\in M,\forall j\in M$$

用更为简洁的方式改写包含足够大容量 K 的上式,得到下式:

$$U(c)=\min\sum_{i=1}^{m}\sum_{j=1}^{n}r_{ij}u_{ij}+\sum_{i=1}^{m}f_iv_i$$

$$\text{s.t.}\ \sum_{i=1}^{m}u_{ij}=1,\forall j\in N$$

$$u_{ij}\le v_i,\forall i\in M,\forall j\in N$$

$$u_{ij},v_i\in\{0,1\},\forall i\in M,\forall j\in N$$

在上述整数规划模型中,u_{ij}表示顾客 j 是否被设施 i 所服务,如果被服务,则 $u_{ij}=1$,否则 $u_{ij}=0$; v_i 表示设施 i 是否开启,如果开启,则 $v_i=1$,否则 $v_i=0$。(1)式为 UFLP 的目标函数,要使得开启设施和运输商品的总成本最小;(2)式表示每个顾客 $j\in N$ 必须被一个设施所服务;(3)式表示如果顾客 j 被设施 i 所服务,则设施 i 必须开启。

逆问题通常有如下的定义,当给定一个加权优化问题和一个固定可行解的实例,其相应的逆问题就是尽可能少地(在一个固定范数下)修改加权参数,使得给定的解在该实例中成为原优化问题的最优解。

根据逆问题的定义,一般优化问题的逆问题有如下形式:

$$\min_d \| d-c \|$$
$$\text{s.t. } d \in G(x^0)$$
$$G(x^0) = \{ d \in \ ^n \mid \min\{ g(d,x) \mid \in D \} = g(d,x^0) \}$$

其中 $g(c,x)$ 为原优化问题的目标函数，$c \in \ ^n$ 为参数向量，D 为 x 的可行域，$d \in \ ^n$ 为调整后的参数向量。$G(x^0)$ 表示使给定的可行解 x^0 成为 $g(d,x^0)$ 最优解的参数向量 d 的集合。

在本文中只对无容量限制设施选址的逆问题进行了讨论。为方便以及更贴近现实，我们以 L_1 范数为例，即表示调整量之和最小。往往对于设施成本和运输成本调整量的权重并不一样，可以分别设为 w_i^f 和 w_{ij}^r。

则 L_1 范数下无容量限制设施选址的逆问题(IUFLP)可以定义如下:

$$\min\ w \times |d-c|^T$$
$$\text{s.t. } dx^0 = U(d)$$
$$d \in \ ^{m+mn}$$

其中 $w=(w^f,w^r)$ 为 $(m+mn)$ 维行向量，$d=(f^0,r^0)$ 为调整成本后的 $(m+mn)$ 维行向量，f^0,r^0 分别表示调整之后的设施成本和运输成本，$c=(f,r)$ 为上述的原成本参数，$|d-c|^T$ 为成本调整量(绝对值)的 $(m+mn)$ 维列向量，x^0 为给定原问题的一个可行解，$U(d)$ 为(1)式中参数为 d 时对应的最小值。

由约束式可以看出，x^0 是满足调整成本后 UFLP 所有约束的最优解。对于每一个顾客 $j \in N$ 可以被任一设施服务，共有 $|M|=m$ 种选择。因而对于所有顾客而言一共会有 m^n 种不同情况。即满足 UFLP 约束条件的可行解有 m^n 个。同时定义满足 UFLP 约束条件的 m^n 个可行解构成的解集为 Y。因此从逆问题的定义出发建立求解模型，Y 中包含可行解的个数是指数个。事实上，无容量限制设施选址的逆问题(IUFLP)是 NP-难的。

定理 1 无容量限制设施选址的逆问题(IUFLP)是 NP-难的。

2 行生成算法求解

从上述无容量限制设施选址逆问题的定义过程中，我们可以看出原优化问题和其逆问题有着一定的联系，即逆问题中的给定解对应的成本需要小于等于所有满足原问题约束条件的解对应的成本。行生成算法作为一种有效求解规模较大线性规划问题的方法。我们可以将行生成算法应用于该问题中，并且容易发现此时的子问题即是 UFLP 原问题，因而行生成算法可以用来求解无容量限制设施选址的逆问题。

无容量限制设施选址的逆问题模型为:

$$\min_{f^0,r^0} \sum_{i=1}^m w_i^f |f_i^0 - f_i| + \sum_{i=1}^m \sum_{j=1}^n w_{ij}^r |r_{ij}^0 - r_{ij}|$$
$$\text{s.t. } f_i^0 v_i^0 + r_{ij}^0 u_{ij}^0 \le f_i^0 v_i' + r_{ij}^0 u_{ij}',\ \forall i \in M,\ \forall j \in N,\ (v',u') \in Y$$

其中 Y 同上述，是满足原问题的所有可行解 $x'=(v',u')$ 的集合。为方便显示我们的结果，我们主要考虑逆最优解问题，即逆问题。而对于逆最优值问题，即给定 V^0 而不是 x^0，根据式 $V^0=cx^0$ 也可用相同思路求解。下文只针对逆问题进行分析求解。

将行生成算法应用于 IUFLP，将 c 对应的设施和运输部分用 (f,r) 分别表示。根据 Ahuja 在 L_1 范数下逆问题的处理方法[5]，记 $c=(f,r)$ 为原成本向量，$d=(f^0,r^0)$ 为调整后的成本向量。令

$$|f_i^0 - f_i| = a_i + b_i,\ |r_{ij}^0 - r_{ij}| = h_{ij} + l_{ij}$$
$$f_i^0 - f_i = a_i - b_i,\ r_{ij}^0 - r_{ij} = h_{ij} - l_{ij}$$

则 a_i,h_{ij} 分别表示相对原成本的增加量，f_i,r_{ij} 表示相对原成本 f_i,r_{ij} 的减少量。

可以得到 IUFLP 的主问题为:

$$(\text{P}) \min_{a,b,h,l} \sum_{i=1}^m w_i^f (a_i + b_i) + \sum_{i=1}^m \sum_{j=1}^m w_{ij}^r (h_{ij} + l_{ij})$$
$$\text{s.t. } (a_i - b_i + f_i)(v_i' - v_i^0) + (h_{ij} - l_{ij} + r_{ij})(u_{ij}' - u_{ij}^0) \ge 0,\ \forall i \in M, \forall j \in N$$
$$a_i \ge 0, b_i \ge 0, \forall i \in M$$
$$h_{ij} \ge 0, l_{ij} \ge 0, \forall i \in M, \forall j \in N$$
$$(v',u') \in \mathbb{I}$$

其中 (v',u') 属于一个可行解限制集 $\mathbb{I} \subset Y$，限制集中的元素由子问题不断生成加入其中。而主问题得到的 (a',b',h',l') 在每次迭代循环中带入到子问题中求解。

子问题如下:

$$(\text{S}) \ T = \min_{v',u'} \sum_{i=1}^m (a_i' - b_i' + f_i')(v_i' - v_i^0) + \sum_{i=1}^m \sum_{j=1}^n (h_{ij}' - l_{ij}' + r_{ij})(u_{ij}' - u_{ij}^0)$$
$$\text{s.t. } \sum_{i=1}^m u_{ij}' = 1, \forall j \in N$$
$$u_{ij}' \le v_i', \forall i \in M, \forall j \in N$$
$$u_{ij}', v_i' \in \{0,1\}, \forall i \in M, \forall j \in N$$

子问题求出得到 (v^*,u^*)，如果目标函数值小于 0，将得到的 (v^*,u^*) 加入到主问题的限制集中。如果目标函数值不小于 0，程序结束。注意到子问题实际上就是 UFLP。

求解 IUFLP 问题的衍生成算法详细步骤如下:

输入: 1) **可行解:** $x^0=(v^0,u^0)$; 2) **原设施成本:** $c=(f,r)$;

输出: 1) **最优解:** (a^*,b^*,h^*,l^*); 2) **最小的总成本改变量:**

$$\sum_{i=1}^{m} w_i^f(a_i^*+b_i^*)+\sum_{i=1}^{m}\sum_{j=1}^{n} w_{ij}^r(h_{ij}^*+l_{ij}^*)$$

3) **对应原成本的改变量:**

$$f_i^0-f_i=a_i^*-b_i^*,r_{ij}^0-r_{ij}=h_{ij}^*-l_{ij}^*$$

初始化: 初始限制集包含对于原问题的可行解即可。为方便且不再重新生成可行解,可取初始限制集为只含给定的可行解这一个元素的集合即 $_0=\{(v^0,u^0)\}$。

步骤 1　求解此时的主问题得到最优解 $(a',b',h'l')$。

步骤 2　将步骤 1 得到的最优解 $(a',b',h'l')$ 带入到子问题中,求解子问题这一整数规划得到最优解 $x^*=(v^*,u^*)$ 和目标函数值 T。

步骤 3　如果目标函数值 $T<0$,将得到的 x^* 加入到限制集中,$\mathrm{Ⅱ}=\mathrm{Ⅱ}\cup\{x^*\}$ 回到步骤 1,重新求解主问题; 如果目标函数值 $T\geq 0$,则结束算法,得到最终的限制集,记为 $\mathrm{Ⅱ}_f$。根据此限制集 $\mathrm{Ⅱ}_f$ 求解主问题,即可得到最优解 (a^*,b^*,h^*,l^*)。

3　启发式算法得到上下界

由上述的衍生成方法,我们可以知道要求得逆问题的最优解就需要得到尽可能精确的主问题的限制集,即主问题中的约束条件。由于主问题的限制集是由子问题不断生成得到的,可以看出衍生成方法的核心是子问题的求解效率,即子问题求解越精确,逆问题的最优解越精确。但由于子问题是设施选址的原问题,因而可以通过对子问题进行松弛,牺牲一定求解精度的情况下简化求解过程。

当对子问题进行如下几种处理方法时,我们可以相应得到衍生成方法的几点结论:

处理方法 1: 严格求解子问题(S)。

定理 2　当严格求解子问题时,循环结束时此时得到 IUFLP 的最优解。

说明: 注意到从子问题中筛选出的解的数量依赖于给定的初始解,虽然衍生成方法可以极大地减少原有的约束,但究竟需要筛选出多少解是无法判断的。事实上,如果可以判断出能够筛选出多少解,根据筛选出的解构成的解集直接求解主问题就可以解决 IUFLP。

处理方法 2: 对子问题利用邻域搜索方法得到启发式整数解,同时设置一定的循环次数提前终止子问题的求解。

定理 3　对子问题利用邻域搜索方法求解,并且设定一定的循环次数提前终止子问题的求解,此时求解得到 IUFLP 的下界。

对于利用邻域搜索求解子问题的说明: 邻域搜索采用基本的本地搜索方法[12],即对于当前给定解采用增加,交换和减少的方式得到局部最优整数解。

下面对邻域搜索的三种方式进行说明。设此时未开启设施的数量为 p,开启设施的数量为 $(m-p)$。**增加,**即对于当前未开启的设施中开启一个设施,作为一个增加邻域,共有 p 个邻域。**交换,**即对于当前未开启的设施中开启一个设施,在开启的设施中关闭一个设施,作为一个交换邻域,共有 $p(m-p)$ 个邻域。**减少,**即对于当前开启的设施中关闭一个设施,作为一个减少邻域,共有 $(m-p)$ 个邻域。

在一次求解过程中,对于三种方式得到的邻域解对应的成本进行比较,取最小值求得对应的局部最优解。将当前给定解转移到得到的局部最优解,同时将该局部最优解加入到主问题的限制集中,再进行下一次求解。

求解 UFLP 下界的启发式算法详细步骤如下:

输入: 1) **可行解:** $x^0=(v^0,u^0)$; 2) **原设施成本:** $c=(f,r)$;

输出: 1) **最优解:** (a^*,b^*,h^*,l^*); 2) **最小的总成本改变量:**

$$\sum_{i=1}^{m} w_i^f(a_i^*+b_i^*)+\sum_{i=1}^{m}\sum_{j=1}^{n} w_{ij}^r(h_{ij}^*+l_{ij}^*)$$

3) **相对原成本的改变量:**

$$f_i^0-f_i=a_i^*-b_i^*,r_{ij}^0-r_{ij}=h_{ij}^*-l_{ij}^*$$

初始化: 初始限制集包含对于原问题的可行解即可。为方便且不再重新生成可行解,可取初始限制集为给定的可行解即 $\mathrm{Ⅱ}=\{(v^0,u^0)\}$。

步骤 1　求解此时的主问题得到最优解 (a',b',h',l')。

步骤 2　将步骤 1 得到的最优解 (a',b',h',l') 带入到子问题中,邻域搜索的子问题目标函数与前述保持一致,对给定的初始解采用增加,交换,减少

操作得到局部最优解 $x^*=(v^*,u^*)$ 和目标函数值 T。

步骤 3 如果目标函数值 $T<0$，将得到的 x^* 加入到限制集中，$\mathrm{II}=\mathrm{II}\cup\{x^*\}$ 回到步骤 1，重新求解主问题；如果目标函数值 $T\geq 0$，则结束算法。此时得到最优解 (a^*,b^*,h^*,l^*)。

其中邻域搜索中的增加，交换，减少操作如图 1 所示：

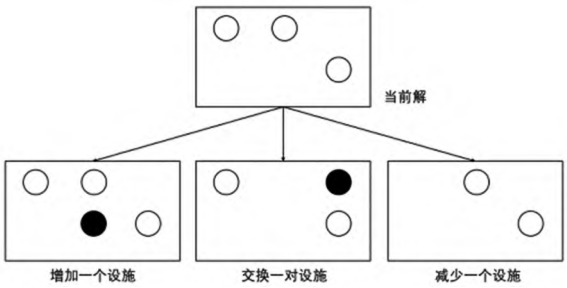

图 1 邻域搜索对当前解进行三种操作的示意图

处理方法 3：对子问题(S)进行线性松弛求解。

定理 4 将子问题(S)进行线性松弛得到线性规划问题，使用行生成算法求解则可以得到对应逆问题的上界。

此时对应的子问题模型变为：

$$T=\min\sum_{i=1}^{m}(a_i'-b_i'+f_i^0)(v_i'-v_i^0)+\sum_{i=1}^{m}\sum_{j=1}^{n}(h_{ij}'-l_{ij}'+r_{ij}^0)(u_{ij}'-u_{ij}^0)$$

$$\text{s.t.}\ \sum_{i=1}^{m}u_{ij}'=1,\forall j\in N$$

$$0\leq u_{ij}'\leq v_i'\leq 1,\forall i\in M,\forall j\in N$$

行生成算法及使用不同处理方法处理子问题求解的流程示意图如下：

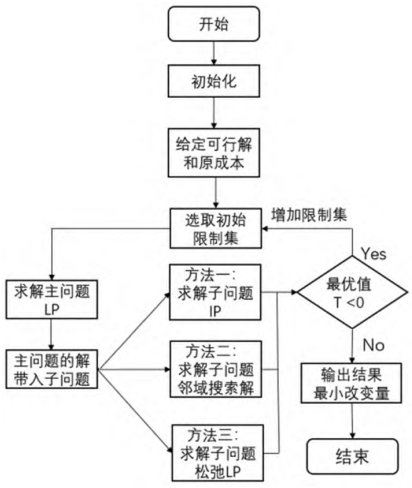

图 2 行生成算法及不同处理方法流程示意图

4 示例及计算结果

4.1 简单的例子

我们举一个简单的例子来具体展示无容量设施选址逆问题。

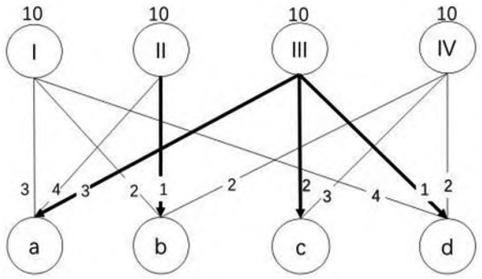

图 3 原 UFLP 示例图

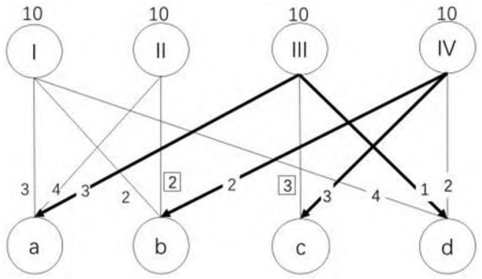

图 4 给定可行解的 IUFLP 示例图

如上图所示，Ⅰ、Ⅱ、Ⅲ、Ⅳ为设施编号，每个设施开启成本为 10。a,c,b,d 为顾客编号，设施与顾客有连线表示该顾客可以被相应设施服务，连线上的数字表示对应的运输成本。黑色的细线表示该问题所有可行的服务连线，带箭头的粗线表示服务最优解连线。相应的设施开启成本和运输成本为：

$$f_i=\begin{pmatrix}10\\10\\10\\10\end{pmatrix},\quad r_{ij}=\begin{pmatrix}3&2&\overline{M}&4\\4&1&\overline{M}&\overline{M}\\3&\overline{M}&2&1\\\overline{M}&2&3&2\end{pmatrix}$$

其中 $\overline{M}$ 表示一个足够大的数。

图 3 中给出了在给定设施开启成本和运输成本后的最优解服务连线。图 4 则是给定了如图所示的服务连线，需要调整设施开启成本或运输成本使得该服务连线成为调整后最优的服务连线。

能够看出，原问题最优解如图 3 所示，即为开设设施Ⅱ服务顾客 b，以及开设设施Ⅲ服务顾客 a,c,d，对应的最优成本为 27。而当给定可行解如图 4 所示时，即为开设设施Ⅲ服务顾客 a,d，以及开设设施Ⅳ服务顾客 b,c 时，对应的成本为 29。需要改变方框中的运输成本使得给定的可行解成为

改变成本后的最优解，即将方框中 1 改为 2，方框中 2 改为 3。此时最小改变量为 2，可以使得给定的可行解成为改变成本后问题的最优解。

4.2　计算结果

本文使用 Windows 10 操作系统，内存 8 GB 处理器为 Intel Core i7-8700 的 PC 电脑上进行了所有的数值实验。所有算法均通过 Matlab R2019a 调用 Gurobi 求解器去实现。在数值实验中，我们对于 6 种不同规模随机生成了对应规模下 20 种可能的实例。$(m,n)=(10,10),(10,20),(10,30),(20,20),(20,40),(30,30)$。

其中设施启动成本和运输成本均为随机产生的整数值，且取值范围分别为:

$f_i=[100,200]$; $r_{ij}=[1,100]$, $[100,200]$

在实际中往往不能改变设施启动成本时，因而在接下来的计算中设置对于设施成本和运输成本调整量的权重分别为 $w^f=\infty$, $w^r=1$。

我们将采用下列方式来比较最优改变量和启发式算法计算得到的改变量之间的差距。当最优改变量不为 0 时，我们采用相对值(上下界改变量-最优改变量)/(最优改变量)的方式来比较 *gap* 的大小。相应的结果见表 1。由于给定的可行解是随机生成的，因而给定可行解刚好是最优解的可能性几乎为零。而当给定可行解刚好是最优解时，根据上述的算法可以知道不管是精确求解算法还是启发式算法，此时的最优改变量均为 0。但在实际问题中，几乎不存在给定的可行解刚好是最优解，因为这样意味着不需要对原有成本做任何的改变，这里不再考虑这一极端情况。

结果如下:

表 1　给定可行解非最优解时，IUFLP 问题的计算结果

规模	OS	T1/s	UB	T2/s	LB	T3/s	GAP1	GAP2
(10,10)-1	1222	0.35	1234	0.28	1215	0.19	0.025	-0.008
(10,20)-1	1381	4.4	1401	3.8	1379	1.7	0.033	-0.002
(10,30)-1	1749	41.8	1775	121	1748	23.8	0.028	-0.001
(20,20)-1	2443	6.0	2453	3.5	2442	2.9	0.015	-0.001
(20,40)-1	2972	606	2988	391	2970	255	0.01	-0.0005
(30,30)-1	3756	55.6	3761	58.3	3755	45.9	0.003	-0.001
(10,10)-2	928	0.7	941	0.5	927	0.4	0.05	-0.001
(10,20)-2	1318	8.1	1334	7.7	1316	2.5	0.021	-0.002
(10,30)-2	1824	41.8	1838	116	1823	22.7	0.014	-0.0005
(20,20)-2	2477	8.9	2487	9.4	2476	3.3	0.007	-0.0004
(20,40)-2	2972	580	2993	322	2969	194	0.013	-0.001
(30,30)-2	3146	78.7	3166	121	3145	44.5	0.02	-0.001

在表中，$(m,n)-1$ 表示 IUFLP 问题的规模。例如 $(20,40)-1$ 表示 20 个设施和 40 个顾客: -1 表示 $f_i=[100,200]$, $r_{ij}=[1,100]$; -2 表示 $f_i=[100,200]$, $r_{ij}=[100,200]$。

OS 表示严格求解子问题时得到的逆问题的最优值。UB 表示使用线性松弛的方法得到的 P 的上界。LB 表示使用邻域搜索得到的 P 的下界。在表 1 中 GAP 计算方法为相对比值法。GAP1 表示 UB 与 OS 之间的 gap，GAP2 表示 LB 与 OS 之间的 gap。T1，T2，T3 分别为得到最优值，上界，下界的计算耗时，单位为秒。表中所有的结果均是一个规模中产生的 20 个实例的平均值，并经过四舍五入处理。

特殊说明: 在使用处理方法二得到下界的过程中，设置了最大循环次数 1000。而在处理方法一和三中，设置的最大循环次数较大不会提前终止程序。

计算结果表明，问题的规模 (m,n) 以及设施开设成本 f_i 与运输成本 r_{ij} 之间的大小关系不会对 gap 的大小造成太大影响。同时，由于该问题是 NP-难的，随着规模的变大，计算复杂度和计算时间都是指数增长的。根据计算结果可以看出，线性松弛得到的上界与最优的改变量差距较小，但求解效率却没有较大提升。原因在于求解最优的改变量时是使用整数规划模型进行求解，而采用的求解器对于整数规划的求解效率较高; 同时线性松弛方法在每次迭代循环过程中产生的是非整数解，迭代循环次数相对变多也增大了求解需要的时间。而利用基本的本地搜索方法求出的下界与最优的改变量非常接近，同时在求解效率上也有较大提升。

因此，当问题规模不大时，可以采用最优解的方式进行求解；而当问题规模较大时，可以采用启发式方法结合一定的迭代循环次数得到不错的下界。迭代循环次数越多，计算耗时就越多，但得到的下界就越好。在实践中，针对具体的问题具体分析，使用启发式方法同时设置一定的迭代循环次数可以得到不错的成本改变量。

5 结论

本文针对无容量限制的设施选址逆问题进行了建模分析，提出了使用行生成算法将逆问题分为了主问题和子问题。通过对子问题中的整数规划模型进行求解可以得到逆问题的最优解。分析表明对于行生成算法中的子问题采用线性松弛法和本地搜索方法可以分别得到逆问题的上下界。数值实验结果表明了线性松弛法得到的上界与最优解较为接近，但求解效率并未有提升；启发式算法在下界的获取上则有着较为优异的表现，同时求解速度较快，说明了该启发式算法的有效性。

实际上，由于子问题是无容量限制设施选址的原问题，可以将求解该问题更为高效的近似算法用于其中，从而提高下界解的质量和求解效率。子问题是最具有优化空间的地方，可以结合启发式算法一次找出多个目标函数小于 0 的解，并一次性全部加入主问题中进行求解，从而提高求解效率。

参考文献：

[1] Mirchandani P B, Francis R L. Discrete location theory [M]. New York: John Wiley and Sons, Inc., 1990.
[2] Jakob K, Pruzan P M. The simple plant location problem: survey and synthesis [J]. European Journal of Operational Research, 1983, 12(1): 36-81.
[3] Tarantola A. Inverse problem theory and methods for model parameter estimation [M]. Society for Industrial and Applied Mathematics, 2005.
[4] Heuberger C. Inverse combinatorial optimization: a survey on problems, methods, and results [J]. Journal of Combinatorial Optimization, 2004, 8(3): 329-361.
[5] Ahuja R K, Orlin J B. Inverse optimization [J]. Operations Research, 2001, 49(5): 771-783.
[6] Cai M C, Yang X G, Zhang J Z. The complexity analysis of the inverse center location problem [J]. Journal of Global Optimization, 1999, 15(2): 213-218.
[7] Schaefer A J. Inverse integer programming [J]. Optimization Letters, 2009, 3(4): 483-489.
[8] Alizadeh B, Burkard R E, Pferschy U. Inverse 1-center location problems with edge length augmentation on trees [J]. Computing, 2009, 86(4): 331.
[9] Guan X, Zhang B. Inverse 1-median problem on trees under weighted hamming distance [J]. Journal of Global Optimization, 2012, 54(1): 75-82.
[10] Berman O, Ingco D I, Odoni A. Improving the location of minimax facilities through network modification [J]. Networks, 1994, 24(1): 31-41.
[11] Zhang J, Liu Z, Ma Z. Some reverse location problems [J]. European Journal of Operational Research, 2000, 124(1): 77-88.
[12] Ghosh D. Neighborhood search heuristics for the uncapacitated facility location problem [J]. European Journal of Operational Research, 2003, 150(1): 150-162.