Numerical Optimization, 2020 Fall

Homework 6

Due on 23:59, Nov. 06, 2020

(NOTE: Homework will not be accepted after this due for any reason.)

1 Warehousees for MX

某西 (MX) 是一个美国邮寄物品租赁公司. 顾客在该公司的网站上租赁物品, 然后 MX 会将它邮寄给顾客. 当顾客完成使用后, 需要将其邮寄回 MX. MX 的商业模式依赖于快速的发货时间 (否则, 顾客没有耐心等待很长时间). 但是, 像某东 (MD) 等公司的半日达速递服务不在我们的问题讨论范围之内, 因为他们的邮寄成本可能会相对较高.

在早期, MX 只有几个配送中心 (Distriution Centers, DC for short), 并且一般需要 2 天的时间才能把物品邮寄到顾客手中. MX 很快意识到提供快速的送货服务对于公司业务提升至关重要, 然后他们决定尝试为大约 90% 的客户提供 1 日达的速递服务. 这项政策在很大程度上导致了几年前利润的大幅增长.

在这道问题中, 您将进行数学建模并求解, 来确定 MX 应该把 DC 放置在何处, 以确保所需的顾客在 1 天邮寄范围内, 同时将建立 DC 的固定成本降至最低. (假定每个 DC 的物品处理和运输的单位成本都是相同的).

a) 将以下问题建模为整数规划 (Integer Programming, IP for short) 问题: 给定一组城市, 以及每个城市的人口和在该城市开设 DC 的固定成本. 目标是决定将 DC 设在哪些城市, 以便将总的固定成本降至最低, 同时还确保至少有 α 人口在 1 天邮寄范围内.

注意: 确保清晰地定义您所使用的符号, 并指明哪些是参数 (输入), 哪些是决策变量. 然后, 使用文字描述清楚您的每一个约束条件. [20pts]

b) 使用 AMPL 实现您创立的模型. 请查看.zip 文件 warehouse.xlsx 中的数据集来解决这个问题. 该数据集提供了全美 150 个大城市的位置和人口数 (根据 2000 年的美国人口普查), 以及在相应城市开设一家 DC 的年均固定成本. 文件中还包含数据集中每对城市之间的距离 (以英里为单位). 若两个城市之间的距离不超过 150 英里, 即可认定为它们在 1 天的邮寄范围内.

另外, 覆盖率 $\alpha = 90\%$, 请找到 MX DC 位置问题的最佳解决方案. 在您提交的文件中须包括两部分: PDF 文件和 .zip 文件 (请您分离提交). 其中模型文件 (.mod), 数据文件 (.dat), 运行

文件 (.run) 以及报告您的解决方案的总成本和要开放的 DC 总数: 用截图的方式提交在 PDF 中; 所提交的 .zip 文件中包含您编写的.mod, .dat, .run 等文件. [25pts]

c) MX 目前大约有 35 个 DC 开放. 这是否接近您在 *b*) 中找到的 DC 数量? 如果不是, 您的解决方案与 MX 的解决方案可能不同的原因是什么? [5pts]

Solution:

a) Define the following notation:

Sets:

$I =$ set of cities

$J =$ set of potential DC locations

Parameters:

$D_s =$ the distance standard mailing in 1-day

$h_i =$ the population in city $i \in I$

$f_j =$ fixed cost to open a DC at $j \in J$

$d_{ij} =$ the travel distance from DC $j \in J$ to city $i \in I$

$a_{ij} =$ is 1 if $d_{ij} \leq D_s$; 0 otherwise

Decision Variables:

$x_i = 1$ if DC $j \in J$ is opened; 0 otherwise

$y_{ij} =$ the fraction of population in city $i \in I$ served by a DC located in $j \in J$

In this problem, we are only concerned with the fixed cost to open the DCs, so the mathematical formulation reads as follows

$$
\begin{aligned}
\min \quad & \sum_{j \in J} f_j x_j \\
\text{s.t.} \quad & \sum_{i \in I} \sum_{j \in J} a_{ij} y_{ij} h_i \geq \alpha \sum_{i \in I} h_i, \\
& \sum_{j \in J} y_{ij} = 1, && \forall i \in I \\
& y_{ij} \leq x_j, && \forall i \in I, \forall j \in J \\
& x_j \in \{0, 1\}, && \forall j \in J \\
& y_{ij} \geq 0, && \forall i \in I, \forall j \in J
\end{aligned}
\tag{1}
$$

In (1), the objective function computes the total fixed cost. The first constraint ensures that at least α fraction of the population is within 1-day mailing range. The second constraints requite the full amount of every city's demand to be assigned. The third constraints prohibit a city from being assigned to a DC that has not been opened (These are often called linking constraints). The fourth constraints require the location variables (x) to be binary, and the last constraints require the assignment variables (y) to be nonnegative.

b) Please refer to the attached .mod, .dat and .run files.

c) Not very close and the number of DCs we found in part *b*) is 23. The possible reasons are that

1. Our formulation does not take transportation costs into account.
2. The DCs have no capacity restrictions-any amount of product can be handled by any DC. Obviously, this is impossible in practice.

2 Solving the MX Problem

在上述问题 **1** 中, 您制定了一个 IP 模型来解决 MX 的 DC 定点问题, 以确保给定的人口分数 (α) 在距离其最近的 DC 的 1 天邮寄范围内. 在此问题中, 您将开发一种基于 Lagrangian 松弛来求解此 IP 的方法.

作为出发点, 您要根据您的 IP 模型松弛某个约束, 并定义一个相应的乘子.

a) 写出这个松弛产生的 Lagrangian 子问题. [20pts]

b) 子问题应分解为两个分离的问题, 一个仅包含 $\boldsymbol{x}$ 变量, 另一个仅包含 $\boldsymbol{y}$ 变量. 写出这两个不同的问题. [20pts]

c) 请解释如何求解两个子问题, 即 $\boldsymbol{x}$-子问题和 $\boldsymbol{y}$-子问题. 您的求解方法可能不依赖于使用单纯型法或任何其他一般的 LP 或 IP 算法. [10pts]

Solution:

a) Consider relaxing the third constraints $y_{ij} \le x_j, \forall i \in I, \forall j \in J$ in (1). Then, the Lagrangian subproblem becomes

$$\begin{aligned} \min \quad & \sum_{j\in J} f_j x_j + \sum_{i\in I}\sum_{j\in J} \lambda_{ij}(x_j - y_{ij}) \\ \text{s.t.} \quad & \sum_{i\in I}\sum_{j\in J} a_{ij} y_{ij} h_i \ge \alpha \sum_{i\in I} h_i, \\ & \sum_{j\in J} y_{ij} = 1, && \forall i \in I \\ & x_j \in \{0,1\}, && \forall j \in J \\ & y_{ij} \ge 0, && \forall i \in I, \forall j \in J \end{aligned} \tag{2}$$

where λ_{ij} is the introduced Lagrange multiplier for the constraints corresponding to city $i \in I$ and node $j \in J$, and it is restricted to be nonpositive.

b) Based on *a*), there are no constraints linking the $\boldsymbol{x}$ and $\boldsymbol{y}$ variables, so the problem can be written as two separate problems:

$$\begin{aligned} \min \quad & \sum_{j\in J}(\sum_{i\in I} \lambda_{ij} + f_j) x_j \\ \text{s.t.} \quad & x_j \in \{0,1\}, && \forall j \in J \end{aligned} \tag{x-problem}$$

$$
\begin{aligned}
\min \quad & -\sum_{i\in I}\sum_{j\in J}\lambda_{ij}y_{ij} & & \\
\text{s.t.} \quad & \sum_{i\in I}\sum_{j\in J}a_{ij}y_{ij}h_i \geq \alpha\sum_{i\in I}h_i, & & \\
& \sum_{j\in J}y_{ij} = 1, & \forall i \in I & \qquad (y\text{-problem}) \\
& y_{ij} \geq 0, & \forall i \in I, \forall j \in J &
\end{aligned}
$$

c) – To solve the x-problem, we simply set $x_j = 1$ for all j such that $\sum_{i\in I}\lambda_{ij} + f_j < 0$.

– To solve the y-problem, for each i we set $y_{ij} = 1$ that minimizes $-\lambda_{ij}$. Note that, each λ gives a single lower (recall that the relaxation problem always produces a lower bound for our original problem) and a single upper bound (e.g., the feasible solutions to the original problem, and it is easy to generate such a feasible solution.). The Lagrangian relaxation process involves many iterations, each using a different value of λ, in the hopes of tightening the bounds. To update the multiplier, we use the subgradient optimization method. At the kth iterate, the step size α^k is given by

$$\alpha^k = \frac{t^k(\text{UB} - \mathcal{L}^k)}{\sum_{i\in I}\sum_{j\in J}(x_j - y_{ij})^2}, \tag{3}$$

where $\mathcal{L}^k$ is the lower bound found at iteration k (i.e., the value of the Lagrangian subproblem for the current value of λ) and UB is the best upper bound found (i.e., the objective value of the best feasible solution found so far-note that while $\mathcal{L}^k$ is the last lower bound found, UB is the best upper bound found). Then the multiplier-updating formula becomes

$$\lambda_{ij}^{k+1} = \lambda_{ij}^k + \alpha^k(x_j - y_{ij}).$$