Localizando Instalaciones

Daniel E. Cordovés Borroto

Email: dcordb97@gmail.com

30 de noviembre de 2020

Índice

1. Introducción **3**
1.1. Formalizando el problema 3
1.2. Sobre la aplicación 3

2. Desarrollo **4**
2.1. FastModel 4
2.2. Resolviendo FastModel 4
2.2.1. Hallando el óptimo 6
2.3. La aplicación 7

3. Conclusiones **8**

1. Introducción

En el siguiente reporte exponemos la solución a un problema de Modelos de Optimización:

- Dado dos conjuntos de instalaciones [1] se desea seleccionar un lugar para crear una nueva instalación de forma tal que esté "cerca" del primer conjunto y "lejos" del segundo.

1.1. Formalizando el problema

El problema tiene una definición informal, especialmente por el uso de palabras como cerca y lejos, por lo que es necesario hacer algunas consideraciones:

1. Consideremos a cada instalación i como un punto (x_i, y_i) en el plano.
2. Denotemos con A al conjunto del cual se quiere estar cerca.
3. Denotemos con B al conjunto del cual se quiere estar lejos.
4. Denotemos a la x del i-ésimo punto de un conjunto S de puntos con $X(S_i)$ y a la y con $Y(S_i)$.
5. Denotemos con (X, Y) al punto solución.
6. Definamos como $f_A(X, Y)$ al valor de cercanía del punto solución al conjunto A.
7. Definamos como $f_B(X, Y)$ al valor de lejanía del punto solución al conjunto B.
8. Definamos como x_1, x_2, y_1, y_2 al rectángulo (cuyo punto inferior izquierdo es (x_1, y_1) y superior derecho (x_2, y_2)) en el cual se puede encontrar el punto solución.
9. Además, de cada instalación i tenemos un peso w_i $(w_i \geq 1)$ que tiene asignado, que influye que tanto la solución se acerque o aleje de esta.

Ahora nuestro problema es buscar un punto (X, Y) tal que maximice $f_A(X, Y)$ y maximice $f_B(X, Y)$. A continuación proponemos dos modelos para la solución de este problema.

1.2. Sobre la aplicación

Debido a la naturaleza geométrica del problema en cuestión, desarrollamos una aplicación visual que permite mostrar la localización de cada instalación en el plano. La aplicación fue desarrollada en Python3 con el uso de las librerías Numpy, Matplotlib, PuLP y PySide2 (Qt).

[1] Por ejemplo: edificios, casas, etc.

2. Desarrollo

Proponemos el modelo FastModel a continuación.

2.1. FastModel

$$\text{mín}\, f_A(X,Y) - f_B(X,Y) \tag{1}$$

Sujeto a:

$$\begin{aligned} f_A(X,Y) &= \sum_{i=1}^{|A|} w(A_i) \cdot \Big[\big|X(A_i) - X\big| + \big|Y(A_i) - Y\big|\Big] \\ f_B(X,Y) &= \sum_{j=1}^{|B|} w(B_j) \cdot \Big[\big|X(B_j) - X\big| + \big|Y(B_j) - Y\big|\Big] \\ x_1 \leq\ & X \leq x_2 \\ y_1 \leq\ & Y \leq y_2 \end{aligned}$$

Este modelo opta por minimizar la diferencia entre la "cercanía" y "lejanía" de las instalaciones al punto solución.

2.2. Resolviendo FastModel

Proposición 1. *Podemos minimizar* (1) *minimizando los términos con* X *y con* Y *de forma independiente.*

Demostración. Si separamos por coordenadas en (1), nos queda:

$$\begin{aligned} S_1(X) &= \sum_{i=1}^{|A|} w(A_i) \cdot \big|X(A_i) - X\big| - \sum_{j=1}^{|B|} w(B_j) \cdot \big|X(B_j) - X\big| \\ S_2(Y) &= \sum_{i=1}^{|A|} w(A_i) \cdot \big|Y(A_i) - Y\big| - \sum_{j=1}^{|B|} w(B_j) \cdot \big|Y(B_j) - Y\big| \end{aligned}$$

por tanto (1) equivale a mín$\{S_1(X) + S_2(Y)\}$, que lo logramos minimizando cada término por separado.

Definición 1. *Para todo* i, j $(1 \leq i \leq |A|, 1 \leq j \leq |B|)$*:*

1. *Un conjunto* T_x *es de puntos originales para la coordenada* x *si* $T_x = \{X(A_i), X(B_j), x_1, x_2\}$.

2. *Un conjunto T_y es de puntos originales para la coordenada y si $T_y = \{Y(A_i), Y(B_j), y_1, y_2\}$.*

Definición 2. *Para todo conjunto S tal que $S' = \{-\infty, \infty\} \cup S$ y $x \in \mathbb{R}$ definimos como:*

1. $\text{ínf}(S, x)$ *al mayor $y \in S'$ tal que $y \leq x$.*
2. $\sup(S, x)$ *al menor $y \in S'$ tal que $y \geq x$.*

Colorario 1. *No existe $t \in S \cap \left[\text{ínf}(S, x), \sup(S, x)\right]$ tal que $t \neq x$.*

Demostración. La demostración es directa por la definición de ínfimo y supremo dada. □

Proposición 2. *Para todo $x \in I$ donde $I = \left[\text{ínf}(T_x, x), \sup(T_x, x)\right]$ se cumple que $S_1(x)$ es una función lineal que depende de x.*

Demostración.

Denotemos a:

- $m(x) = \{X(A_i) \mid X(A_i) < x\} \cup \{X(B_j) \mid X(B_j) < x\}$
- $M(x) = \{X(A_i) \mid X(A_i) > x\} \cup \{X(B_j) \mid X(B_j) > x\}$

Por el Colorario 1 sabemos que $m(x)$ y $M(x)$ no cambian en I. Lo que nos permite quitarnos el valor absoluto de las sumas. Al hacer esto, agrupamos los términos con y sin x, obteniendo una función lineal. □

Proposición 3. *Para todo $x \in \mathbb{R}$, $p = \text{ínf}(T_x, x)$ y $q = \sup(T_x, x)$ se cumple que $\text{mín}\{S_1(p), S_1(q)\} \leq S_1(x)$.*

Demostración. Usando la Prop. 2 sabemos que $S_1(x)$ se puede expresar como $m \cdot x + n$.

Analicemos tres casos:

1. Si $m > 0 \implies S_1(x)$ se minimiza en p.
2. Si $m < 0 \implies S_1(x)$ se minimiza en q.
3. Si $m = 0 \implies S_1(x)$ se minimiza en cualquier $t \in [p, q]$ (en particular en p o en q).

Por lo que queda demostrado. □

Proposición 4. *Existe un punto solución* (X, Y)[2] *tal que* $X \in T_x$ *y* $Y \in T_y$.

Demostración.

Por la Prop. 1 sabemos que la coordenada x es independiente de la y, lo que nos permite minimizar X y Y por separado. Sin pérdida de generalidad demostremos que X es solución, ya que Y es un caso simétrico.

Supongamos que X no pertenece a T_x, entonces existen números p y q tal que $p, q \in T_x$ y $p \leq X \leq q$; en particular se cumple que $p = \text{ínf}(T_x, X)$ y $q = \sup(T_x, X)$.

Luego por la Prop. 3 nos encontramos en una contradicción, dado que encontramos un óptimo con valor menor o igual a $S_1(X)$ que sí pertenece a T_x.

Por lo que X tiene que pertenecer a T_x. □

Ahora por la Prop. 4 nuestro problema se reduce a evaluar cada elemento del conjunto T_x y T_y en S_1 y S_2 respectivamente.

2.2.1. Hallando el óptimo

Vamos a mostrar como evaluar $S_1(X)$, ya que $S_2(Y)$ es simétrico.

En primer lugar ordenamos el conjunto A por su coordenada x. Denotemos como L al miembro izquierdo de $S_1(X)$ y como R al miembro derecho.

Definimos p como la menor posición i $(1 \leq i \leq |A|)$ tal que $X(A_i) > X$. Entonces se cumple que:

$$\begin{aligned} L_1 &= X \cdot (w(A_1) + \ldots + w(A_{p-1})) - X(A_1) \cdot w(A_1) - \ldots - X(A_{p-1}) \cdot w(A_{p-1}) \\ L_2 &= -X \cdot (w(A_p) + \ldots + w(A_{|A|})) + X(A_p) \cdot w(A_p) + \ldots + X(A_{|A|}) \cdot w(A_{|A|}) \\ L &= L_1 + L_2 \end{aligned}$$

En otras palabras, las coordenadas x de las instalaciones i $(1 \leq i < p)$ son todas menores o iguales a X y el resto son mayores, por lo que podemos quitarnos el valor absoluto. Para R nos queda algo similar. Notar que esto es lo que nos queda luego de desarrollar la demostración de la Def. 2 (solo teniendo en cuenta L).

Ahora solo tenemos que fijar un $X \in T_x$, hay $O(n)$ de ellos, donde $n = |T_x| = |A| + |B| + 2$; y hallar L y R. Dado un X fijo podemos hallar el costo correspondiente en $O(\log n)$, porque solo tenemos que buscar p con una búsqueda binaria; y para hallar el costo podemos precalcular sumas acumulativas que nos van a permitir hallar la suma en un rango en $O(1)$. [3] Por lo que podemos resolver el modelo en $O(n \log n)$.

[2] Notar que el modelo tiene las restricciones $x_1 \leq X \leq x_2$ y $y_1 \leq Y \leq y_2$.
[3] Más detalles sobre esto se pueden ver en el código.

2.3. La aplicación

La aplicación nos permite entrar los conjuntos A y B; además de los límites x_1, x_2, y_1, y_2 y graficarlos. Una vez graficados es posible interactuar con el gráfico haciendo zoom o moviendo las coordenadas visibles. También permite guardar y cargar los valores entrados a la aplicación para una futura visualización.

Debido a que esto es un reporte más bien teórico, no ponemos las instrucciones de la aplicación acá. Estas se pueden encontrar en el botón de ayuda de la propia aplicación.

3. Conclusiones

Se cumplieron los objetivos planteados:

- Presentamos un modelo para resolver el problema inicial.
- Desarrollamos una aplicación para el análisis y la visualización de las posiciones de las instalaciones.