Modelo Matematico para la Acomodación de un Basurero en una Ciudad.

Maria Andrea Cruz Blandon 0831816 Edgar Andres Moncada Taborda 0832294
Hebert Vargas Tello 1124798 Luis Felipe Vargas Rojas 0836342

4 de enero de 2013

Índice

1. Introducción

En el siguiente documento se explicará el modelo contruido a través del paradigma de la programacion linea (lp), para resolver el problema de la acomodación de un basurero en una ciudad, el problema se basa en que la construcción del basurero se debe realizar en el lugar más alejado posible de la ciudad mas cercana, esto debido a los malos olores que genera y a las molestias de cada ciudad por dicha construcción.

Los principales problemas a los que nos enfentramos al tratar de modelar este problema con lp fuerón definir los dominios las variables, establecer una forma para definir distancias entre ciudades y el basurero a través de un valor absoluto, definir a traves de restricciones la ciudad más cercana entre otros detalles que se explicaran en el documento.

Para la implementación del modelo usamos el programa lpsolve y una libreria de java que genera el archivo de entrada para el lpsolve teniendo en cuenta los valores de entrada.

2. Modelo Programación Lineal

2.1. Datos de Entrada

T	Tamaño de la grilla (T x T)
N	Numero de ciudades.
X_i	Coordenada x de la ciudad i.
Y_i	Coordenada y de la ciudad i.

2.2. Variables del Problema

Xb	Coordenada x del basurero
Yb	Coordenada y del basurero
Xc	Coordenada x de la ciudad más cercana al basurero
Yc	Coordenada y de la ciudad más cercana al basurero
Zx_i	Diferencia entre la coordenada x del basurero (Xb) y la coordenada x de la ciudad i
Zy_i	Diferencia entre la coordenada y del basurero (Yb) y la coordenada y de la ciudad i
Zx_{ia}	Usada para el valor absoluto de la diferencia entre entre la ciudad i y el basurero representa la parte positiva de la variable Zx_i
Zx_{ib}	Usada para el valor absoluto de la diferencia entre la ciudad i y el basurero, representa la parte negativa de la variable Zx_i
Zy_{ia}	Usada para el valor absoluto de la diferencia entre la ciudad i y el basurero representa la parte positiva de la variable Zy_i
Zy_{ib}	Usada para el valor absoluto de la diferencia entre la ciudad i y el basurero representa la parte negativa de la variable Zy_i
dx	Diferencia entre Xc - Xb *Usado para la función objetivo*
dy	Diferencia entre Yc - Yb *Usado para la función objetivo*
Zx	Valor absoluto de dx
Zy	Valor absoluto de dy

2.3. Variables Binarias del Problema

Usadas para el Valor Absoluto de la Diferencia de cada Ciudad con el Basurero:

Bx_{ia}	Usada en las restricciones del valor absoluto para la ciudad i coordenada x
Bx_{ib}	Usada en las restricciones del valor absoluto para la ciudad i coordenada x
By_{ia}	Usada en las restricciones del valor absoluto para la ciudad i coordenada y
By_{ib}	Usada en las restricciones del valor absoluto para la ciudad i coordenada y

Usadas para que $Xc \in X_i$ y $Yc \in Y_i$.

Bc_i	Toma el valor de 1 si la ciudad i es la más cercana

Usadas para el valor absoluto de la función objetivo.

$Bx0$	Usada para el valor absoluto en la función objetivo coordenada x
$By0$	Usada para el valor absoluto en la función objetivo coordenada y

2.4. Restricciones Obvias

Dominios de las Variables

$$0 \leq Xb, Yb, Xc, Yc,\ Zx,\ Zy \leq T$$

Zx_i , Zy_i Son variables irrestrictas.

2.5. Restricciones del Problema

Garantizar que: $Xc \in X_i$ y $Yc \in Y_i$.

$$Xc = \sum_i X_i * Bc_i$$
$$Yc = \sum_i Y_i * Bc_i$$
$$\sum_i Bc_i = 1$$
$$i \in [1, N]$$

En la implementación se debe normalizar las restricciones por eso la sumatoria pasa al otro lado a restar y queda igual a cero la restricción, esto se hace con todas las restricciones.

Diferencias de distancias entre el basurero y la ciudad i, Zx_i,Zy_i

$$Zx_i = X_i - Xb$$
$$Zy_i = Y_i - Yb$$
$$i \in [1, N]$$

Valor absoluto para Zx_i,Zy_i

$$Zx_i = Zx_{ia} - Zx_{ib}$$
$$Zy_i = Zy_{ia} - Zy_{ib}$$

$$M * Bx_{ia} - Zx_{ia} >= 0$$
$$M * Bx_{ib} - Zx_{ib} >= 0$$

$$M * By_{ia} - Zy_{ia} >= 0$$
$$M * By_{ib} - Zy_{ib} >= 0$$

$$Bx_{ia} + Bx_{ib} = 1$$
$$By_{ia} + By_{ib} = 1$$

$$i \in [1, N]$$
$$M = T + 1$$

$$|Zx_i| = Zx_{ia} + Zx_{ib}$$
$$|Zy_i| = Zy_{ia} + Zy_{ib}$$

Para que Zx,Zy **tomen los valores de la ciudad mas cercana al basurero se debe cumplir que:**

$$|Zx_i| + |Zy_i| \geq Zx + Zy$$
$$i \in [1, N]$$

Diferencia entre ciudad más cercana y basurero:

$$dx = Xc - Xb$$
$$dy = Yc - Yb$$

2.6. Función Objetivo.

Para la función objetivo nos encontramos de nuevo con el problema del valor absoluto ya que calculamos diferencias y podemos obtener resultados con valores negativos.

La función objetivo definida es:

$$\text{MAX}(\ Zx + Zy)$$

Donde Zx y Zy son la representación en valor absoluto de dx y dy, respectivamente por lo cual esta función objetivo esta sujeta a :

$$
\begin{gathered}
dx + M * B_{x0} - Zx \geq 0 \\
dx + M * B_{x0} + Zx \leq M \\
dx \leq Zx \\
-dx \leq Zx \\
M = T * 2
\end{gathered}
$$

$$
\begin{gathered}
dy + M * B_{y0} - Zy \geq 0 \\
dy + M * B_{y0} + Zy \leq M \\
dy \leq Zy \\
-dy \leq Zy
\end{gathered}
$$

3. Implementación

Para la implementación del modelo presentado anteriormente usamos el programa lpSolver, lpSolver es un programa para resolver problemas de programación lineal y programación entera.

LpSolver es un programa de codigo abierto licencia LGPL basado en el metodo simplex para resolver problemas de programación lineal y en el metodo branch and bound para resolver problemas de programación entera.

Nuestro protecto fue contruido con el ID netbeans, los metodos basicos que usamos de la libreria fueron :

```
LpSolve lp= LpSolve.makeLp(0, Ncol);
```

Función que instancia el lp se le define el numero de columnas (variables).

```
lp.addConstraintex(Ncol, row, colno, LpSolve.LE, sizeGrid);
```

Función que crea la restricción, row es un arreglo con los coeficientes de las variables ingresadas en el arreglo colno LpSolve.LE hace parte de las definiciones de las condiciones mayor igual, menor o igual, igual.

```
lp.setBinary(posVarBinary, true);
```

Se le indica al solver que la variable ubicada en la columna posVarBinary es binaria.

```

lp.setUnbounded(j + 1);
```

Se le indica al solver que la variable ubicada en la posición j+1 es irrestricta

```
lp.setColName(1, "Zx");
```

Función que nos permite asignarle un nombre a la variable para una mejor visualización.

```

lp.setObjFnex(Ncol, row, colno);
lp.setMaxim();
```

Definimos cuales son las variables involucradas en la función objetivo , luego definimos que debemos maximizar esa función