PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA
Departamento de Ingeniería Industrial y de Sistemas

Curso: ICS1113-Optimización
Ayudante: Javiera Gebhardt
Mail: javieragr@uc.cl

Taller Python - Gurobi

Parte 1. Estructura básica de un modelo en Python-Gurobi

```
from gurobipy import GRB, Model
from gurobipy import quicksum #para las sumatorias

#----------------------- Generacion del modelo ------------------------

model = Model()
model.setParam("TimeLimit", 60) #Establece el tiempo máximo en segundos

#---------------- Se instancian variables de decision -----------------

# vtype : tipo de variable
# Variables binarias: vtype = GRB.BINARY
# Variables continuas: vtype = GRB.CONTINUOUS
# Variables enteras: vtype = GRB.INTEGER

# Para definir solamente a UNA variable:
x = model.addVar(vtype = GRB.INTEGER, name = "x")
#Para definir un conjunto de variables que dependen del conjunto T:
y = model.addVars(T, vtype = GRB.CONTINUOUS, name = "y")

#------------------ Agregar las variables al modelo -------------------

model.update()

#----------------------- Agregar Restricciones ------------------------

# Para agregar solamente UNA restriccion:
model.addConstr(restriccion, name="R1")
#Para definir un conjunto de restricciones
model.addConstrs(conjunto_restricciones, name="R2")

#------------------------ Funcion Objetivo ---------------------------

model.setObjective(funcion_objetivo, GRB.MAXIMIZE) # o GRB.MINIMIZE
model.optimize()

#------------------------ Manejo Soluciones ---------------------------

# Valor objetivo:
valor_objetivo = model.ObjVal

# Solucion para una variable que no depende de conjuntos:
print(f"La variable x toma el valor de {x.x}")
```

```python
# Solucion para variables que dependen de conjuntos:
for t in T:
    print(f"Para t = {t} la variable y toma el valor de {y[t].x}")
# Mostrar los valores de todas las soluciones que no son cero
model.printAttr("X")

#---------------------------- Holguras ------------------------------
# Imprime las holguras de las restricciones
# (0 significa que la restricción es activa).
for constr in model.getConstrs():
    print(constr, constr.getAttr("slack"))
```

Parte 2. Problema básico

Resolver el siguiente modelo básico con el uso de Python - Gurobi

$$
\begin{array}{llrcrr}
(P) & \max & 2x+4y+5z & & & \\
& \text{s.a.} & 3x+y+2z & \leq & 8 & (1) \\
& & x+2y+z & \leq & 10 & (2) \\
& & -x-y-3z & \geq & -14 & (3) \\
& & x,y,z & \geq & 0 & (4),(5),(6)
\end{array}
$$

Resultado: $(x, y, z) = (0, 4, 2)$

Parte 3. Pregunta 1 Tarea 2 2021-1

El municipio de una comuna rural de Chile ha decidido levantar campamentos de vacunación para llegar a la mayor cantidad de habitantes. La comuna se puede dividir en un conjunto I localidades, donde en cada una viven p_i personas con $i \in I$. Análisis preliminares realizandos en la comuna han establecido que los campamentos solo pueden ser ubicados en un conjunto J de sitios predeterminados dentro de la comuna. El municipio debe seleccionar los sitios en los cuales levantar los campamentos (en un sitio cabe a lo más un campamento) considerando que se debe asignar un campamento a cada localidad. Es decir, cada localidad de la comuna debe tener uno (y solo un) campamento de vacunación asociado, mientras que un campamento puede estar asociado a más de una localidad. Construir un campamento en el sitio $j \in J$ tiene un costo fijo asociado igual a c_j. Además, existe un costo variable que es linealmente proporcional (con pendiente f_j) a la cantidad total de personas que debe servir el campamento. Es decir, si se construye un campamento en el sitio $j \in J$, entonces el costo asociado es $c_j + f_j s_j$, donde s_j es la población total que debe servir el campamento ubicado en $j \in J$ (es la suma de las poblaciones de todas las localidades asociadas a ese campamento).

El objetivo es minimizar la distancia máxima entre una localidad y su respectivo campamento de vacunación sin incurrir en un costo superior a P donde $d_{ij} > 0$ es la distancia desde el centro de la localidad $i \in I$ al sitio $j \in J$. El problema puede modelarse como:

Variables de decisión:

- $\diamond$ $x_j : \begin{cases} 1 & \text{Si se construye en un campamento en el sitio j} \in J \\ 0 & \text{En otro caso.} \end{cases}$
- $\diamond$ $y_{ji} : \begin{cases} 1 & \text{Si se asocia la localidad } i \in I \text{ con el campamento ubicado en el sitio } j \in J \\ 0 & \text{En otro caso.} \end{cases}$
- $\diamond$ s_j: cantidad de personas asignadas para vacunarse en el campamento construido en el sitio $j \in J$.

$$
\begin{array}{rcll}
 & \text{mín} & w & \\
s.a: & & & \\
w & \geq & d_{ij} y_{ij} & \forall i \in I, j \in J \\
x_j & \geq & y_{ij} & \forall i \in I, j \in J \\
\sum_{j \in J} y_{ij} & = & 1 & \forall i \in I \\
s_j & \geq & \sum_{i \in I} y_{ij} p_i & \forall j \in J \\
\sum_{j \in J} x_j c_j + s_j f_j & \leq & P & \\
x_j & \in & \{0,1\} & \forall j \in J \\
y_{ij} & \in & \{0,1\} & \forall i \in I, j \in J \\
s_j & \geq & 0 & \forall j \in J
\end{array}
$$

Utilizando Python + Gurobi, encuentre la solución óptima de la instancia adjunta.

Parte 4. Instalación de Estaciones de Bomberos

Se acaba de inaugurar una nueva comuna en un país vecino; una de las decisiones que se deben tomar es dónde instalar las dos estaciones de bomberos necesarias. La comuna ha sido dividida en 5 distritos distintos, y no se puede localizar más de una bomba en cada distrito. Cada bomba debe atender los incendios del distrito donde esté instalada, así como los incendios que se le asignen de otros distritos. Las decisiones involucradas son: en qué distrito instalar las estaciones, y las asignaciones de los restantes distritos a las estaciones. El objetivo es minimizar el tiempo promedio de respuesta a todos los incendios, considerando que solo se pueden instalar dos bombas, que cada distrito debe ser atendido por una bomba y que si en el distrito i no se ha instalado ninguna bomba no es posible atender a ningun otro distrito desde ahí. La Tabla 1 muestra los tiempos promedios de respuesta a un incendio por cada distrito y ubicación posible de una estación; además de incluir el número promedio de incendios diarios en cada uno de ellos.

	Incendio en Distrito				
Estación ubicada en	**1**	**2**	**3**	**4**	**5**
1	5	12	30	20	15
2	20	4	15	10	25
3	15	20	6	15	12
4	25	15	25	4	10
5	10	25	15	12	5
Promedio Incendios	2	1	3	1	3

Tabla 1: Tiempos de Respuesta a Incendios y Promedio de Incendios

El modelo de programación lineal entera para resolver este problema se presenta a continuación:

Variables de decisión:

- $\diamond\ x_i : \begin{cases} 1 & \text{Si se instala una bomba en el distrito } i \\ 0 & \text{En otro caso.} \end{cases}$

- $\diamond\ z_{ji} : \begin{cases} 1 & \text{Si el distrito } j \text{ es atendido por la bomba ubicada en el distrito } i \text{ con } i \neq j \\ 0 & \text{En otro caso.} \end{cases}$

$$
\begin{array}{rccl}
\text{mín} & Min \sum_{i=1}^{5} \sum_{j=1}^{5} n_j t_{ij} z_{ij} & & \\
s.a: & & & \\
\sum_{i=1}^{5} x_i & = & 2 & \\
\sum_{i=1, i \neq j}^{5} z_{ij} & = & 1 & \forall j \\
z_{ij} & \leq & x_i & \forall i \neq j \\
x_i & \in & \{0,1\} & \forall j \\
z_{ij} & \in & \{0,1\} & \forall i, j
\end{array}
$$

Utilizando Python + Gurobi, encuentre la solución óptima.

Parte 5. Más ejemplos de modelos...

Pueden ver más ejemplos de modelos en el siguiente link : Ejemplos modelos Python + Gurobi