Maestría en Ciencia de Datos e Información

“Optimización de costos en la asignación de ingenieros a solicitudes de usuario”

Proyecto Final

Matemáticas para la Ciencia de Datos

Equipo 902-E:

Ismael Medina Muñoz

Helio Jiménez García

Resumen

El objetivo del presente trabajo es encontrar la combinación óptima de asignaciones entre un conjunto de ingenieros calificados y una cantidad de solicitudes de servicio a clientes. El término de “optimo” se refiere a cumplir con las solicitudes de servicio de cada cliente, en el tiempo que se requiere y asignando un ingeniero capaz de resolver el problema, todo esto al menor costo posible.

El objetivo general es plantear este problema como un modelo de programación lineal mediante el cual podamos encontrar la combinación óptima de ingenieros y solicitudes de servicio que minimicen el costo total. Desde luego que el objetivo también abarca la solución de este problema utilizando un algoritmo de asignación.

Una vez que el modelo se plantee de manera correcta y se alcance una solución que satisfaga los requerimientos de la administración, este proceso se puede implementar repetidamente de manera rápida y efectiva, para la planeación de la fuerza de trabajo.

1. Introducción

Las empresas disponen de recursos que, aplicados de manera correcta, la distinguen de otras empresas similares; dentro de dichos recursos el más importante es el personal del que dispone la empresa.

Uno de los objetivos empresariales permanentes a través del tiempo ha sido como aprovechar al máximo los recursos disponibles y desde luego incluimos el recurso humano porque en algunos casos es uno de los gastos más importantes para la organización.

El presente trabajo busca aplicar el conocimiento científico para la mejor asignación de trabajos a un grupo de empleados en una empresa de software. Es decir, la empresa cuenta con un grupo de ingenieros calificados que son llamados a atender las necesidades de servicio de la cartera de clientes con los que cuenta la empresa. La particularidad de esta situación es que por lo regular la cantidad de servicios o reparaciones es mayor a la cantidad de ingenieros con los que la compañía cuenta y la Dirección de la empresa busca aprovechar al máximo los recursos humanos y al mismo tiempo cubrir las expectativas de calidad de servicio y tiempo de entrega para cada uno de los clientes.

La complejidad de la situación se radica en el hecho de que cada solicitud de servicio requiere un nivel de especialización que algunos ingenieros poseen mientras que otros poseerán especializaciones que no son adecuadas para atender otros servicios. Un segundo aspecto que aumenta la complejidad es que cada ingeniero tiene una agenda que hace patente la necesidad de empatar el calendario de espacios de tiempo disponible de cada ingeniero con el periodo de tiempo que cada solicitud de servicio tiene asignada.

Dadas las características del problema, donde tenemos recursos limitados que compiten por la realización de ciertas actividades, una resolución desde el enfoque de la Investigación de Operaciones, en específico la Programación Lineal Entera, resulta en una primera opción para generar una solución. Aquí entonces consideramos que es un problema de optimización de recursos de un proceso de la actividad humana. En específico deseamos minimizar los costos generados por la atención de los ingenieros sobre las solicitudes de servicio de los clientes.

En esencia este trabajo busca implementar una solución ya conocida dentro del mundo de la Programación Lineal, denominada modelo de asignación, a un problema real sacado del mundo empresarial de Latinoamérica y encontrar la mejor solución a una problemática a la que los administradores se enfrentan de manera cotidiana.

2. Planteamiento del problema

En una empresa de servicios especializados entregados por ingenieros calificados, los ingenieros viajan a la localidad de los clientes para la entrega de estos. La entrega de cada servicio sólo dura una semana, por lo que la planeación de asignaciones se hace con esta cadencia. Los ingenieros tienen una matriz de acreditaciones para la entrega de servicios, por lo que, dependiendo de la solicitud de los clientes, sólo algunos ingenieros estarían acreditados para entregar el servicio solicitado.

Dado que los servicios se solicitan en toda Latinoamérica y que los ingenieros están localizados en ciertos países, un ingeniero con una acreditación en el servicio solicitado cerca del país donde el cliente se ubica puede ser una primera noción de la posibilidad de una reducción de costos bajo ciertas restricciones.

Los ingenieros tienen una agenda muy ocupada y los clientes buscan recibir sus servicios lo antes posible. Por política de la empresa, cada cliente debe estar siendo visitado por un ingeniero antes de dos meses.

Matriz de solicitudes

A continuación, se muestra un ejemplo de la tabla de solicitudes de servicio que se han realizado a la compañía por diversos clientes.

IdServicio	IdCliente	Fecha de la solicitud
1	10	2021-06-07
1	12	2021-06-14
4	3	2021-05-24
…	…	…
2	9	2021-06-07

Con estas solicitudes, los científicos de datos armaron una matriz que contiene el número de solicitudes del servicio en el lapso de las siguientes 8 semanas (2 meses aproximadamente) por cada cliente.
Para acotar el problema, se tomaron sólo los clientes con los contratos más grandes y que pueden comprar más servicios, incrementando las ganancias del negocio.

	IdServicio					
IdCliente	**01**	**02**	**03**	**04**	**05**	**06**
1	**2**	**0**	**0**	**0**	**0**	**0**
2	**0**	**0**	**1**	**0**	**0**	**0**
3	**1**	**1**	**0**	**0**	**0**	**0**
4	**0**	**0**	**0**	**0**	**2**	**0**
5	**0**	**0**	**0**	**1**	**0**	**1**

La matriz de acreditaciones de ingenieros

La empresa tiene un registro de las acreditaciones de sus ingenieros con un identificador binario. Estos se muestran en la siguiente tabla

Id_Ingeniero	IDServ 01	IDServ 02	IDServ 03	IDServ 04	IDServ 05	IDServ 06
1	0	0	1	1	0	1
2	0	0	1	0	1	1
3	1	1	0	1	0	0
4	0	1	0	0	1	1
5	0	0	0	1	0	1
6	1	1	1	1	0	0

La matriz de la agenda de ingenieros

La empresa tiene un registro de las semanas que un ingeniero ya tiene asignaciones. Un ejemplo de los datos existentes se puede encontrar en la siguiente tabla:

IdIngeniero	IdServicio	Fecha de inicio asignación
1	10	2021-06-07
1	12	2021-06-14
4	3	2021-05-24
…	…	…
2	9	2021-06-07

Con estas asignaciones, los científicos de datos armaron una matriz que contiene el número de semanas libres del ingeniero en el lapso de las siguientes 8 semanas (2 meses aproximadamente). Tenemos 6 ingenieros contratados.

ID_Ingeniero	Semanas Libres
1	2
2	1
3	3
4	2
5	1
6	3

La matriz de costos de entrega de servicios por cliente e ingenieros

La empresa tiene un registro de los gastos esperados de enviar un ingeniero a las instalaciones de cada cliente sin importar el servicio. De esta manera, los científicos de datos pueden asociar costos en las asignaciones para así minimizar los gastos de viaje de un ingeniero a algún servicio en particular. Aquí se anexa la tabla de los costos en USD de tener un ingeniero asignado una semana.

IdIngeniero	IdCliente 01	IdCliente 02	IdCliente 03	IdCliente 04	IdCliente 05
1	120	113	91	97	60
2	68	97	68	127	113
3	111	115	120	114	119
4	93	113	113	93	112
5	108	67	115	115	114
6	125	107	60	113	64

Con los datos proporcionados, defina la asignación de ingenieros que minimice el costo de los servicios de los clientes pero que todos los clientes tengan ingeniero asignado para ejecutar el ser. Si es posible, busque que todos los clientes tengan asignaciones de ingenieros para entregar los servicios solicitados.

En este trabajo se propone encontrar una solución a la problemática planteada basados en realidades científicas que nos garanticen la mejor solución posible para las condiciones dadas y evitar la utilización de métodos de prueba y error que son tardados y no garantizan la mejor solución posible.

Dadas las características planteadas anteriormente este caso se ajusta a un problema de optimización del tipo de “asignación” que se resuelve utilizando la teoría de programación lineal entera.

3. Marco teórico

Dada la problemática expuesta con anterioridad, utilizaremos el modelo de programación lineal entera de asignación, ya que debemos de empatar recursos limitados (ingenieros) con actividades competidoras (los clientes).

El modelo de asignación clásico trata de hacer coincidir a los ingenieros (con diferentes habilidades) con los clientes. Presumiblemente, la variación de habilidades afecta el costo de completar un trabajo. El objetivo es determinar el costo mínimo de asignación de ingenieros a los clientes que requieren servicio. El modelo de asignación general con n trabajadores y n puestos de trabajo se representa en la figura 1. El elemento c_{ij} representa el costo de asignar al ingeniero i al cliente j $(i, j=1,2,\ldots,n)$. No hay pérdida de generalidad al suponer que el número de ingenieros y el número de clientes son iguales, porque siempre podemos agregar trabajadores o trabajos ficticios para satisfacer este supuesto. (Taha, 2017)

El modelo de asignación es un caso especial del modelo de transporte donde los trabajadores representan fuentes y los trabajos representan destinos. La cantidad de oferta (demanda) en cada fuente (destino) es exactamente igual a 1. El costo de "transportar" al trabajador i al trabajo j es c_{ij}. En efecto, el modelo de asignación se puede resolver directamente como un modelo de transporte regular (o como un problema de PL regular) (Taha, 2017).

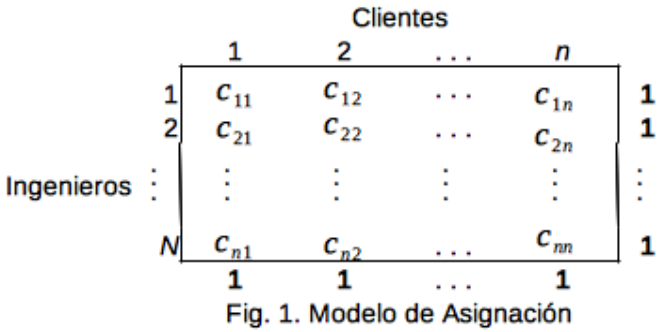

Fig. 1. Modelo de Asignación

Fig. 1 Modelo de Asignación

El problema de asignación en el cual n trabajadores son asignados a n trabajos puede ser representado como un modelo de programación lineal de la siguiente manera. Sea c_{ij} el costo de asignar el trabajador i al trabajo j, y definimos a:

$$x_{ij}=\begin{cases}1, & \text{si el trabajador } i \text{ es asignado al trabajo } j\\ 0, & \text{en cualquier otro caso}\end{cases}$$

Entonces el modelo de programación lineal esta dado por:

Minimizar $Z=\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij}$

Sujeto a las siguientes restricciones:

$$\sum_{j=1}^{n} x_{ij}=1, i=1,2,\ldots,n$$

$$\sum_{i=1}^{n} x_{ij}=1, j=1,2,\ldots,n$$

$$x_{ij}=0 \acute{o} 1$$

4. Objetivos

1. Plantear el problema descrito como un modelo de programación lineal entera del tipo de asignación.
2. Resolver el modelo de optimización y obtener el mínimo costo de asignación de ingenieros a los clientes que han solicitado servicio.

5. Metodología

La empresa cuenta con un conjunto limitado de ingenieros especializados que deben atender solicitudes de servicio para los clientes. La cantidad de clientes que deben ser atendidos por lo general supera la cantidad de ingenieros disponibles para un periodo cualquiera.

Existen una serie de restricciones que aplican tanto para los ingenieros disponibles como para la atención de los clientes. Los ingenieros cuentan cada uno, con ciertas acreditaciones que reflejan el nivel de habilidad y experiencia para cada uno de ellos, por lo tanto, cada uno se diferencia por su nivel de acreditación que viene asociado a un costo individualizado.

Los clientes también cuentan con una serie de características que los hacen diferentes, como la distancia a la que se encuentran de la planta de los ingenieros, el tipo de cliente y el nivel de problema presentado.

La problemática se centra en encontrar la asignación optima de ingenieros a los clientes que necesitan un servicio de tal forma que el problema del cliente se resuelva de manera definitiva y se minimicen los costos totales del servicio.

A continuación se presenta la programación del modelo utilizando las herramientas de programación computacional Python & Sagemath, que se han utilizado para transformar el problema real en un modelo de programación lineal y posteriormente en un modelo computacional para su ejecución y solución.

5.1. Declaración de estructuras de datos, matriz de solicitudes

Se crean las estructuras de datos necesarias para la ejecución del programa.

A continuación se muestra la tabla de clientes vs servicios. Cada cliente hace una o varias solicitudes de servicios y estos se clasifican en la matriz mostrada mas abajo. Nos ayuda a conocer que cliente requiere que tipo de servicio y en que cantidad.

```
======================  PROGRAMA EN PYTHON & SAGEMATH  ======================

import pandas as pd

data = [['C1', 2, 0, 0, 0, 0, 0],
        ['C2', 0, 0, 1, 0, 0, 0],
        ['C3', 1, 1, 0, 0, 0, 0],
        ['C4', 0, 0, 0, 0, 2, 0],
        ['C5', 0, 0, 0, 1, 0, 1]]

solicitudes = pd.DataFrame(data, columns=['IdCliente', 'Servicio01', 'Servicio02',
                  'Servicio03', 'Servicio04', 'Servicio05', 'Servicio06'])


solicitudes = solicitudes.set_index(['IdCliente'])
solicitudes
```

Tabla1. Solicitudes de servicio

IdCliente	Servicio 01	Servicio 02	Servicio 03	Servicio 04	Servicio 05
C1	2	0	0	0	0
C2	0	0	1	0	0
C3	1	1	0	0	0
C4	0	0	0	0	2
C5	0	0	0	1	0

5.2 Declaración de la matriz de acreditaciones

En la siguiente tabla se muestra la matriz de acreditaciones de los Ingenieros, es decir, que ingeniero puede atender que servicio. La atención de cada servicio dependa de la dificultad de la solicitud y de la experiencia y habilidades del ingeniero.

```
data = [['I1', 0, 0, 1, 1, 0, 1],
        ['I2', 0, 0, 1, 0, 1, 1],
        ['I3', 1, 1, 0, 1, 0, 0],
        ['I4', 0, 1, 0, 0, 1, 1],
        ['I5', 0, 0, 0, 1, 0, 1],
        ['I6', 1, 1, 1, 1, 0, 0]]


acreditaciones = pd.DataFrame(data, columns=['IdIngeniero', 'Servicio01',
                  'Servicio02', 'Servicio03', 'Servicio04', 'Servicio05',
                  'Servicio06'])

acreditaciones = acreditaciones.set_index(['IdIngeniero'])
acreditaciones
```

Tabla 2. Acreditaciones de Ingenieros

IdIngeniero	Servicio 01	Servicio 02	Servicio 03	Servicio 04	Servicio 05	Servicio 06
I1	0	0	1	1	0	1
I2	0	0	1	0	1	1
I3	1	1	0	1	0	0
I4	0	1	0	0	1	1
I5	0	0	0	1	0	1
I6	1	1	1	1	0	0

5. 3. Matriz de semanas libres

La siguiente tabla nos muestra la cantidad de semanas libres que cada ingeniero dispone para atender las solicitudes de los clientes.

```
data = [['I1', 2],
        ['I2', 1],
        ['I3', 3],
        ['I4', 2],
        ['I5', 1],
        ['I6', 3]]

libres = pd.DataFrame(data, columns=['IdIngeniero', 'SemanasLibres'])
```

```
libres = libres.set_index(['IdIngeniero'])
libres
```

Tabla 3. Semanas Libres

IdIngeniero	Semanas Libres
I1	2
I2	1
I3	3
I4	2
I5	1
I6	3

5.4. Matriz de costos

En la siguiente tabla observamos los costos asociados a cada ingeniero cuando atiende al servicio de ciertos clientes. Estos costos dependen de varios factores como el salario del ingeniero, los gastos de viaje asociados, la complejidad del servicio, etc.

```
data = [['I1', 120, 113,  91,  97,  60],
        ['I2',  68,  97,  68, 127, 113],
        ['I3', 111, 115, 120, 114, 119],
        ['I4',  93, 113, 113,  93, 112],
        ['I5', 108,  67, 115, 115, 114],
        ['I6', 125, 107,  60, 113,  64]]

costos = pd.DataFrame(data, columns=['IdIngeniero', 'C1', 'C2', 'C3', 'C4', 'C5'])

costos = costos.set_index(['IdIngeniero'])
costos
```

Tabla 4. Costos

IdIngeniero	C1	C2	C3	C4	C5
I1	120	113	91	97	60
I2	68	97	68	127	113
I3	111	115	120	114	119
I4	93	113	113	93	112

I5	108	67	115	115	114
I6	125	107	60	113	64

5. 5. Definición del modelo de programación lineal entera.

En esta sección del código podemos observar la programación del modelo en el lenguaje Sagemath que está optimizado para la solución de modelos de programación lineal.

En primera instancia se define el tipo de programa a utilizar. En este caso estamos utilizando programación lineal entera y con un enfoque de minimización de costos.

```
programa = MixedIntegerLinearProgram(maximization=False)
asignacion = programa.new_variable(integer=True, nonnegative = True,
name='Asignación')
```

5. 6. Programación de la Función Objetivo

Posteriormente viene la sección de la programación de la Función Objetivo, donde le definimos al programa que costos asociamos con las combinaciones de Ingenieros a clientes, esta función se debe de minimizar, es decir el programa buscará el costo mínimo asociado a las combinaciones de ingeniero - cliente.

```
suma = 0
for i in solicitudes.index:
    if solicitudes.Servicio01.loc[i] > 0:
        print("El cliente", i, "requiere", solicitudes.Servicio01.loc[i],
"servicios. Los ingenieros posibles son:")
        for j in acreditaciones.index:
            if acreditaciones.Servicio01.loc[j] > 0:
                print("\tEl ingeniero" , j, "con costo de $", costos[i].loc[j])
                suma = suma + costos[i].loc[j] * asignacion[i, j]
    if solicitudes.Servicio02.loc[i] > 0:
        print("El cliente", i, "requiere", solicitudes.Servicio02.loc[i],
"servicios. Los ingenieros posibles son:")
        for j in acreditaciones.index:
            if acreditaciones.Servicio02.loc[j] > 0:
```

```
                print("\tEl ingeniero" , j, "con costo de $", costos[i].loc[j])
                suma = suma + costos[i].loc[j] * asignacion[i, j]
    if solicitudes.Servicio03.loc[i] > 0:
        print("El cliente", i, "requiere", solicitudes.Servicio03.loc[i],
"servicios. Los ingenieros posibles son:")
        for j in acreditaciones.index:
            if acreditaciones.Servicio03.loc[j] > 0:
                print("\tEl ingeniero" , j, "con costo de $", costos[i].loc[j])
                suma = suma + costos[i].loc[j] * asignacion[i, j]
    if solicitudes.Servicio04.loc[i] > 0:
        print("El cliente", i, "requiere", solicitudes.Servicio04.loc[i],
"servicios. Los ingenieros posibles son:")
        for j in acreditaciones.index:
            if acreditaciones.Servicio04.loc[j] > 0:
                print("\tEl ingeniero" , j, "con costo de $", costos[i].loc[j])
                suma = suma + costos[i].loc[j] * asignacion[i, j]
    if solicitudes.Servicio05.loc[i] > 0:
        print("El cliente", i, "requiere", solicitudes.Servicio05.loc[i],
"servicios. Los ingenieros posibles son:")
        for j in acreditaciones.index:
            if acreditaciones.Servicio05.loc[j] > 0:
                print("\tEl ingeniero" , j, "con costo de $", costos[i].loc[j])
                suma = suma + costos[i].loc[j] * asignacion[i, j]
    if solicitudes.Servicio06.loc[i] > 0:
        print("El cliente", i, "requiere", solicitudes.Servicio06.loc[i],
"servicios. Los ingenieros posibles son:")
        for j in acreditaciones.index:
            if acreditaciones.Servicio06.loc[j] > 0:
                print("\tEl ingeniero" , j, "con costo de $", costos[i].loc[j])
                suma = suma + costos[i].loc[j] * asignacion[i, j]

suma = suma - (costos['C3'].loc['I3']*asignacion['C3','I3'] +
costos['C3'].loc['I6']*asignacion['C3','I6'] +
costos['C5'].loc['I1']*asignacion['C5','I1'] +
costos['C5'].loc['I5']*asignacion['C5','I5'])

programa.set_objective(suma)
programa.show()
```

Desglose de la función objetivo.

```
El cliente C1 requiere 2 servicios. Los ingenieros posibles son:
    El ingeniero I3 con costo de $ 111
    El ingeniero I6 con costo de $ 125
El cliente C2 requiere 1 servicios. Los ingenieros posibles son:
    El ingeniero I1 con costo de $ 113
    El ingeniero I2 con costo de $ 97
    El ingeniero I6 con costo de $ 107
El cliente C3 requiere 1 servicios. Los ingenieros posibles son:
    El ingeniero I3 con costo de $ 120
```

```
    El ingeniero I6 con costo de $ 60
El cliente C3 requiere 1 servicios. Los ingenieros posibles son:
    El ingeniero I3 con costo de $ 120
    El ingeniero I4 con costo de $ 113
    El ingeniero I6 con costo de $ 60
El cliente C4 requiere 2 servicios. Los ingenieros posibles son:
    El ingeniero I2 con costo de $ 127
    El ingeniero I4 con costo de $ 93
El cliente C5 requiere 1 servicios. Los ingenieros posibles son:
    El ingeniero I1 con costo de $ 60
    El ingeniero I3 con costo de $ 119
    El ingeniero I5 con costo de $ 114
    El ingeniero I6 con costo de $ 64
El cliente C5 requiere 1 servicios. Los ingenieros posibles son:
    El ingeniero I1 con costo de $ 60
    El ingeniero I2 con costo de $ 113
    El ingeniero I4 con costo de $ 112
    El ingeniero I5 con costo de $ 114
```

Definición de la Función Objetivo en Sagemath.

```
Minimization:
  111.0 Asignación[('C1', 'I3')] + 125.0 Asignación[('C1', 'I6')]
+ 113.0 Asignación[('C2', 'I1')] + 97.0 Asignación[('C2', 'I2')]
+ 107.0 Asignación[('C2', 'I6')] + 120.0 Asignación[('C3', 'I3')]
+ 60.0 Asignación[('C3', 'I6')] + 113.0 Asignación[('C3', 'I4')]
+ 127.0 Asignación[('C4', 'I2')] + 93.0 Asignación[('C4', 'I4')]
+ 60.0 Asignación[('C5', 'I1')] + 119.0 Asignación[('C5', 'I3')]
+ 114.0 Asignación[('C5', 'I5')] + 64.0 Asignación[('C5', 'I6')]
+ 113.0 Asignación[('C5', 'I2')] + 112.0 Asignación[('C5', 'I4')]

Constraints:
Variables:
 Asignación[('C1', 'I3')] = x_0 is an integer variable (min=0.0, max=+oo)
 Asignación[('C1', 'I6')] = x_1 is an integer variable (min=0.0, max=+oo)
 Asignación[('C2', 'I1')] = x_2 is an integer variable (min=0.0, max=+oo)
  Asignación[('C2', 'I2')] = x_3 is an integer variable (min=0.0, max=+oo)
  Asignación[('C2', 'I6')] = x_4 is an integer variable (min=0.0, max=+oo)
  Asignación[('C3', 'I3')] = x_5 is an integer variable (min=0.0, max=+oo)
  Asignación[('C3', 'I6')] = x_6 is an integer variable (min=0.0, max=+oo)
  Asignación[('C3', 'I4')] = x_7 is an integer variable (min=0.0, max=+oo)
  Asignación[('C4', 'I2')] = x_8 is an integer variable (min=0.0, max=+oo)
  Asignación[('C4', 'I4')] = x_9 is an integer variable (min=0.0, max=+oo)
  Asignación[('C5', 'I1')] = x_10 is an integer variable (min=0.0, max=+oo)
  Asignación[('C5', 'I3')] = x_11 is an integer variable (min=0.0, max=+oo)
  Asignación[('C5', 'I5')] = x_12 is an integer variable (min=0.0, max=+oo)
  Asignación[('C5', 'I6')] = x_13 is an integer variable (min=0.0, max=+oo)
  Asignación[('C5', 'I2')] = x_14 is an integer variable (min=0.0, max=+oo)
  Asignación[('C5', 'I4')] = x_15 is an integer variable (min=0.0, max=+oo)
```

5. 7. Definición de las restricciones del programa

En esta parte las restricciones están asociadas a que cada ingeniero debe de ser asignado a un solo cliente.

```
programa.add_constraint(asignacion[('C2', 'I1')] + asignacion[('C5', 'I1')] <= 2)
programa.add_constraint(asignacion[('C2', 'I2')] + asignacion[('C4', 'I2')] +
asignacion[('C5', 'I2')] <= 1)
programa.add_constraint(asignacion[('C1', 'I3')] + asignacion[('C3', 'I3')] +
asignacion[('C5', 'I3')] <= 3)
programa.add_constraint(asignacion[('C3', 'I4')] + asignacion[('C4', 'I4')] +
asignacion[('C5', 'I4')] <= 2)
programa.add_constraint(asignacion[('C5', 'I5')] <= 1)
programa.add_constraint(asignacion[('C1', 'I6')] + asignacion[('C2', 'I6')] +
asignacion[('C3', 'I6')] + asignacion[('C5', 'I6')] <= 3)
```

En esta parte de las restricciones, nos aseguramos que a cada cliente se la asigne un solo ingeniero para la solución de los servicios que ha solicitado.

```
programa.add_constraint(asignacion[('C1', 'I3')] + asignacion[('C1', 'I6')] == 2)
programa.add_constraint(asignacion[('C2', 'I1')] + asignacion[('C2', 'I2')] +
asignacion[('C2', 'I6')] == 1)
programa.add_constraint(asignacion[('C3', 'I3')] + asignacion[('C3', 'I6')] +
asignacion[('C3', 'I4')] == 2)
programa.add_constraint(asignacion[('C4', 'I2')] + asignacion[('C4', 'I4')] == 2)
programa.add_constraint(asignacion[('C5', 'I1')] + asignacion[('C5', 'I3')] +
asignacion[('C5', 'I5')] + asignacion[('C5', 'I6')] + asignacion[('C5', 'I2')] +
asignacion[('C5', 'I4')] == 2)

programa.show()
```

5.8. Definición completa del modelo en Sagemath.

A continuación podemos ver la programación completa de nuestro problema de programación lineal.

```
Minimization:
  111.0 Asignación[('C1', 'I3')] + 125.0 Asignación[('C1', 'I6')] + 113.0
Asignación[('C2', 'I1')] + 97.0 Asignación[('C2', 'I2')] + 107.0 Asignación[('C2',
'I6')] + 120.0 Asignación[('C3', 'I3')] + 60.0 Asignación[('C3', 'I6')] + 113.0
Asignación[('C3', 'I4')] + 127.0 Asignación[('C4', 'I2')] + 93.0 Asignación[('C4',
'I4')] + 60.0 Asignación[('C5', 'I1')] + 119.0 Asignación[('C5', 'I3')] + 114.0
Asignación[('C5', 'I5')] + 64.0 Asignación[('C5', 'I6')] + 113.0 Asignación[('C5',
'I2')] + 112.0 Asignación[('C5', 'I4')]

Constraints:
  Asignación[('C2', 'I1')] + Asignación[('C5', 'I1')] <= 2.0
  Asignación[('C2', 'I2')] + Asignación[('C4', 'I2')]
+ Asignación[('C5', 'I2')] <= 1.0
 Asignación[('C1', 'I3')] + Asignación[('C3', 'I3')]
+ Asignación[('C5', 'I3')] <= 3.0
 Asignación[('C3', 'I4')] + Asignación[('C4', 'I4')]
+ Asignación[('C5', 'I4')] <= 2.0
 Asignación[('C5', 'I5')] <= 1.0
Asignación[('C1', 'I6')] + Asignación[('C2', 'I6')]
+ Asignación[('C3', 'I6')] + Asignación[('C5', 'I6')] <= 3.0

2.0 <= Asignación[('C1', 'I3')] + Asignación[('C1', 'I6')] <= 2.0
 1.0 <= Asignación[('C2', 'I1')] + Asignación[('C2', 'I2')] + Asignación[('C2',
'I6')] <= 1.0
 2.0 <= Asignación[('C3', 'I3')] + Asignación[('C3', 'I6')] + Asignación[('C3',
'I4')] <= 2.0
 2.0 <= Asignación[('C4', 'I2')] + Asignación[('C4', 'I4')] <= 2.0
 2.0 <= Asignación[('C5', 'I1')] + Asignación[('C5', 'I3')] + Asignación[('C5',
'I5')] + Asignación[('C5', 'I6')] + Asignación[('C5', 'I2')] + Asignación[('C5',
'I4')] <= 2.0

Variables:
  Asignación[('C1', 'I3')] = x_0 is an integer variable (min=0.0, max=+oo)
  Asignación[('C1', 'I6')] = x_1 is an integer variable (min=0.0, max=+oo)
  Asignación[('C2', 'I1')] = x_2 is an integer variable (min=0.0, max=+oo)
  Asignación[('C2', 'I2')] = x_3 is an integer variable (min=0.0, max=+oo)
  Asignación[('C2', 'I6')] = x_4 is an integer variable (min=0.0, max=+oo)
  Asignación[('C3', 'I3')] = x_5 is an integer variable (min=0.0, max=+oo)
  Asignación[('C3', 'I6')] = x_6 is an integer variable (min=0.0, max=+oo)
  Asignación[('C3', 'I4')] = x_7 is an integer variable (min=0.0, max=+oo)
  Asignación[('C4', 'I2')] = x_8 is an integer variable (min=0.0, max=+oo)
  Asignación[('C4', 'I4')] = x_9 is an integer variable (min=0.0, max=+oo)
  Asignación[('C5', 'I1')] = x_10 is an integer variable (min=0.0, max=+oo)
  Asignación[('C5', 'I3')] = x_11 is an integer variable (min=0.0, max=+oo)
  Asignación[('C5', 'I5')] = x_12 is an integer variable (min=0.0, max=+oo)
  Asignación[('C5', 'I6')] = x_13 is an integer variable (min=0.0, max=+oo)
  Asignación[('C5', 'I2')] = x_14 is an integer variable (min=0.0, max=+oo)
  Asignación[('C5', 'I4')] = x_15 is an integer variable (min=0.0, max=+oo)
```

Solución de la función objetivo.

En esta sección se presenta la solución general de la función objetivo, que en nuestro caso es minimizar los costos.

“El costo mínimo total de asignación de los ingenieros a los clientes con las condiciones definidas en el problema es de 745 dolares”.

```
solucion = programa.solve()
print(solucion)
```

$745.0

5.9. Desglose de asignaciones Ingeniero – Cliente.

En esta sección del código se hace un desglose de la atención de los servicios por cada uno de los ingenieros

```
suma = 0
previouskey = ''
for key, val in programa.get_values(asignacion).items():
    ##if val != 0:
    solicitudestotal = solicitudes.Servicio01.loc[key[0]] +
solicitudes.Servicio02.loc[key[0]] + solicitudes.Servicio03.loc[key[0]] +
solicitudes.Servicio04.loc[key[0]] + solicitudes.Servicio05.loc[key[0]] +
solicitudes.Servicio06.loc[key[0]]
    if previouskey != key[0]:
        print("El cliente", key[0] , "solicitó", solicitudestotal, "servicios")
        previouskey = key[0]
    parcial = val * costos[key[0]].loc[key[1]]
    suma += parcial
    print("\tEl ingeniero", key[1] , "dara {:.0f}".format(val), "servicios a costo
de ${:,.2f}".format(parcial))
print("\nEl costo del programa es ${:,.2f}".format(suma))
```

El cliente C1 solicitó 2 servicios
El ingeniero I3 dará 2 servicios a costo de $222.00
El ingeniero I6 dará 0 servicios a costo de $0.00

El cliente C2 solicitó 1 servicios
El ingeniero I1 dará 0 servicios a costo de $0.00
El ingeniero I2 dará 1 servicios a costo de $97.00
El ingeniero I6 dará 0 servicios a costo de $0.00

El cliente C3 solicitó 2 servicios
El ingeniero I3 dará 0 servicios a costo de $0.00
El ingeniero I6 dará 2 servicios a costo de $120.00
El ingeniero I4 dará 0 servicios a costo de $0.00

El cliente C4 solicitó 2 servicios
El ingeniero I2 dará 0 servicios a costo de $0.00
El ingeniero I4 dará 2 servicios a costo de $186.00

El cliente C5 solicitó 2 servicios
El ingeniero I1 dará 2 servicios a costo de $120.00
El ingeniero I3 dará 0 servicios a costo de $0.00
El ingeniero I5 dará 0 servicios a costo de $0.00
El ingeniero I6 dará 0 servicios a costo de $0.00
El ingeniero I2 dará 0 servicios a costo de $0.00
El ingeniero I4 dará 0 servicios a costo de $0.00

El costo del programa es $745.00

6. Análisis de resultados

El problema de asignación de ingenieros calificados a solicitudes de clientes, se ha planteado como un problema de programación lineal entera del tipo de asignación.

Se han utilizado las herramientas de programación Python & Sagemath que contienen paquetes de código específicamente diseñados para la solución de modelos como los antes descritos.

Las herramientas de programación computacional requieren de los datos necesarios para la ejecución de las subrutinas de optimización, de tal forma que la información es suministrada en forma de tablas de datos. La información necesaria para la solución del problema esta representada en la matriz de solicitudes de servicio, la matriz de acreditaciones de los ingenieros y la matriz de costos de entrega del servicio.

Para encontrar la solución del problema se requiere la especificación de una función objetivo y un conjunto de restricciones que limitan el resultado de minimización de costos.

En la sección 5.6 podemos observar la programación en código Sagemath de la función objetivo, donde especificamos que deseamos minimizar los costos que resultan de asignar un conjunto posible de ingenieros a las solicitudes de servicio pendientes que los clientes requieren. Se programan todas los combinaciones posibles y el programa cuando quede solucionado, seleccionará aquellas combinaciones con menor costo.

En la sección 5.7 se realiza la programación de las restricciones del problema, básicamente son de dos tipos, asegurarnos que los clientes reciban el servicio a todas y cada una de sus solicitudes y por otra parte, que los ingenieros no rebasen la cantidad de servicios que pueden ejecutar.

Tras la ejecución del modelo completo por la herramienta Sagemath, se obtiene el resultados del costo total de asignar a los ingenieros a los requerimientos de servicios. En este caso el costo mínimo por cumplir con todos las solicitudes de servicio es de $745.00 dolares. Este costo es el mínimo posible que podemos alcanzar bajo las restricciones que hemos declarado, este dato lo podemos observar en l sección 5.8.

En la sección 5.9 podemos observar el desglose de este costo mínimo, con el detalle de asignación de ingenieros a clientes y cual es el costo por asignación individual.

También se puede observar que todos los ingenieros han sido asignados a una o mas solicitudes, excepto el ingeniero 5 que no tiene solicitudes por atender.

Este resultado nos garantiza un costo mínimo para la empresa en la asignación de ingenieros a clientes. Sin embargo, si se hace estrictamente necesario que todos los ingenieros ejecuten un servicio a la semana, entonces nos tendríamos que desviar de este planteamiento óptimo y asignar un trabajo al ingeniero 5 con la consecuencia de incrementar los costos para la empresa.

7. Referencias bibliográficas

1. Hamdy A. Taha. (2017). *Operations Research an Introduction, 10th Edition*. Pearson Education.

8. Cronograma

Etapa	Inicio	Fin
Selección del problema	18/05/2021	20/05/2021
Definición del problema	20/05/2021	22/05/2021
Definición de objetivos	22/05/2021	24/05/2021
Metodología a utilizar	24/05/2021	26/05/2021
Implementación del modelo	27/05/2021	30/05/2021
Validación de resultados	30/05/2021	01/06/2021
Interpretación de resultados	01/06/2021	01/06/2021
Documentación	18/05/2021	01/06/2021