UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA INDUSTRIAL

Tarea 1

Otoño 2022

IN3171-1 – Modelamiento y Optimización

Integrantes:	Antonia Arias C.
	Tomas Diaz Y.
	Vicente Cornejo R.
Profesor:	Matías Romero Y.
	Benjamín Barrientos F.
Auxiliares:	Alonso Martínez V.
	Bruno Hernández P.
	Camilo Escalante L.
	Catalina Leppe S.
	Claudia Navarro A.
	Daniela Sanchez Z.
	Eduardo J. Silva G.
	Florencia Vargas D.
	Ignacio Cantillano V.
	Javiera Núñez P.
	M. Jesús Vargas
	Monserrat Marchant S.M.
	Natalia G. González

Fecha de entrega: 02 de Abril de 2022
Santiago, Chile

Índice de Contenidos

Índice de Figuras

1. Introducción

- En el presente informe se expone el desarrollo de 2 tipos de problemas de modelamiento matemático y la búsqueda de su solución óptima. Estos problemas son variaciones de los conocidos "problema del vendedor viajero", el cual busca minimizar el tiempo del recorrido que debe hacer el vendedor pasando por todas las ciudades disponibles y volviendo a su punto de origen, y el problema de "la mochila" que busca maximizar el beneficio de llevar cosas en un objeto con una determinada capacidad. Para esto, con la ayuda de la teoría, se genera un modelamiento mediante Variables de decisión y sus respectivas restricciones para, finalmente, plantear una función objetivo la cual debe ser optimizada. Esto se realiza con la ayuda de la programación en *Phyton*, desarrollando un algoritmo que represente el modelamiento realizado para cada problema, y su programa *Gurobi* el cual permite conectar el algoritmo con la optimización.

  Estos problemas permiten conocer la forma de llevar una encrucijada de la vida real a un planteamiento matemático y su solución.

2. Problema 1

A partir de los parámetros expuestos en la pregunta 1 se obtienen los siguientes planteamientos.

2.1. Variable de decisión

- X_{ij} = 1, si la persona i ∈ N asume el rol de presentador en la reunión j ∈ J 0, si no
- Y_{ij} = 1, si la persona i ∈ N asume el rol de evaluador en la reunión j ∈ J 0, si no
- Z_{ij} = 1, si la persona i ∈ N asume el rol de anotador en la reunión j ∈ J 0, si no
- O_{ij} = 1, si la persona i ∈ N asume el rol de oyente en la reunión j ∈ J 0, si no.

Donde $J = 1, \ldots, N$ e $I = 1, \ldots, M$ con N y M valores constantes conocidos.

2.2. Restricciones

Observación: estas restricciones servirán para las todas las partes que componen esta pregunta.

1. "Naturaleza de la variable"
   $X_{ij}, Y_{ij}, Z_{ij}, O_{ij} \in \{0, 1\}, \forall i \in I\ \forall j \in J$
2. "1 rol por persona por cada reunión"
   $\sum_{i \in I} \mathrm{X}_{ij} + Y_{ij} + Z_{ij} + O_{ij} = 1\ \forall j \in J$
3. "Todos deben participar en 1 rol al menos 1 vez"
   $\sum_{j \in J} \mathrm{X}_{ij} = 1\ \forall i \in I$
   $\sum_{j \in J} \mathrm{Y}_{ij} = 1\ \forall i \in I$
   $\sum_{j \in J} \mathrm{Z}_{ij} = 1\ \forall i \in I$
4. "Cantidad de roles por reunión"
   $\sum_{i \in I} \mathrm{X}_{ij} \geq m_p\ \forall j \in J$
   $\sum_{i \in I} \mathrm{Y}_{ij} \geq m_e\ \forall j \in J$
   $\sum_{i \in I} \mathrm{Z}_{ij} \geq m_a\ \forall j \in J$

2.3. Función Objetivo

- Para esta parte de la pregunta se define la siguiente función objetivo:

$$max \sum_{i \in I} \sum_{j \in J} X_{ij} + Y_{ij} + Z_{ij} + O_{ij} \tag{1}$$

Esta función tiene por objetivo maximizar el compromiso el cual se mide como la cantidad de roles que asume cada persona luego de las N reuniones.

- Para esta parte de la pregunta se define la siguiente función objetivo:

$$min \sum_{i\in I}\sum_{j\in J} t_p X_{ij} + t_e Y_{ij} + t_a Z_{ij} \tag{2}$$

  Esta función no se considera al rol de oyente, ya que este no aporta "tiempo" en el desarrollo de la reunión, y además tiene por objetivo minimizar el tiempo total dedicado por los roles en cada reunión, a partir de las constantes t_p, t_e, t_a las que representan las horas invertidas por una persona al sumir un rol.

- Para esta parte de la pregunta se define la siguiente función objetivo:

$$max \sum_{i\in I}\sum_{j\in J} X_{ij}(T - t_p) + Y_{ij}(T - t_e) + Z_{ij}(T - t_a) \tag{3}$$

  Al igual que en la función anterior no se considera el rol de oyente por la misma razón.

  Esta función busca maximizar la eficiencia de cada reunión a partir de una estimación de un tiempo T promedio que aporta cada persona al asumir un rol en cada reunión. Desde un punto de vista, esta función busca maximizar la "ganancia" (tiempo que aporta cada persona al asumir un rol en una reunión) bajo un "costo" (tiempo que pierde cada persona en asumir un rol en una reunión).

  Finalmente, al programar el problema de programación lineal se obtienen los siguientes resultados:

  Roles que asume la persona: 1 [1, 'rol que asume=anotador', 2, 'rol que asume=anotador', 3, 'rol que asume=anotador', 4, 'rol que asume=entrevistador', 5, 'rol que asume=presentador']

  Roles que asume la persona: 2 [1, 'rol que asume=entrevistador', 2, 'rol que asume=anotador', 3, 'rol que asume=anotador', 4, 'rol que asume=presentador', 5, 'rol que asume=anotador']

  Roles que asume la persona: 3 [1, 'rol que asume=anotador', 2, 'rol que asume=anotador', 3, 'rol que asume=entrevistador', 4, 'rol que asume=presentador', 5, 'rol que asume=anotador']

  Roles que asume la persona: 4 [1, 'rol que asume=anotador', 2, 'rol que asume=entrevistador', 3, 'rol que asume=anotador', 4, 'rol que asume=anotador', 5, 'rol que asume=presentador']

  Roles que asume la persona: 5 [1, 'rol que asume=anotador', 2, 'rol que asume=presentador', 3, 'rol que asume=anotador', 4, 'rol que asume=entrevistador', 5, 'rol que asume=anotador']

  Roles que asume la persona: 6 [1, 'rol que asume=entrevistador', 2, 'rol que asume=anotador', 3, 'rol que asume=anotador', 4, 'rol que asume=anotador', 5, 'rol que asume=presentador']

  Roles que asume la persona: 7 [1, 'rol que asume=anotador', 2, 'rol que asume=presentador', 3, 'rol que asume=anotador', 4, 'rol que asume=anotador', 5, 'rol que asume=entrevistador']

  Roles que asume la persona: 8 [1, 'rol que asume=presentador', 2, 'rol que asume=entrevistador', 3, 'rol que asume=anotador', 4, 'rol que asume=anotador', 5, 'rol que asume=anotador']

  Roles que asume la persona: 9 [1, 'rol que asume=anotador', 2, 'rol que asume=entrevistador', 3, 'rol que asume=presentador', 4, 'rol que asume=anotador', 5, 'rol que asume=anotador']

  Roles que asume la persona: 10 [1, 'rol que asume=anotador', 2, 'rol que asume=presentador', 3, 'rol que asume=entrevistador', 4, 'rol que asume=anotador', 5, 'rol que asume=anotador']

Roles que asume la persona: 11 [1, 'rol que asume=entrevistador', 2, 'rol que asume=anotador', 3, 'rol que asume=presentador', 4, 'rol que asume=anotador', 5, 'rol que asume=anotador']

Roles que asume la persona: 12 [1, 'rol que asume=anotador', 2, 'rol que asume=anotador', 3, 'rol que asume=anotador', 4, 'rol que asume=presentador', 5, 'rol que asume=entrevistador']

Roles que asume la persona: 13 [1, 'rol que asume=anotador', 2, 'rol que asume=anotador', 3, 'rol que asume=presentador', 4, 'rol que asume=entrevistador', 5, 'rol que asume=anotador']

Roles que asume la persona: 14 [1, 'rol que asume=presentador', 2, 'rol que asume=anotador', 3, 'rol que asume=anotador', 4, 'rol que asume=anotador', 5, 'rol que asume=entrevistador']

Roles que asume la persona: 15 [1, 'rol que asume=entrevistador', 2, 'rol que asume=anotador', 3, 'rol que asume=presentador', 4, 'rol que asume=anotador', 5, 'rol que asume=anotador']

Valor optimo= 30.0

```
Set parameter Username
Academic license - for non-commercial use only - expires 2023-03-06
Gurobi Optimizer version 9.5.1 build v9.5.1rc2 (win64)
Thread count: 4 physical cores, 4 logical processors, using up to 4 threads
Optimize a model with 135 rows, 300 columns and 750 nonzeros
Model fingerprint: 0xb7c191ea
Variable types: 0 continuous, 300 integer (300 binary)
Coefficient statistics:
  Matrix range     [1e+00, 1e+00]
  Objective range  [1e+00, 1e+00]
  Bounds range     [1e+00, 1e+00]
  RHS range        [1e+00, 3e+00]
Found heuristic solution: objective -0.0000000
Presolve removed 0 rows and 75 columns
Presolve time: 0.25s
Presolved: 135 rows, 225 columns, 675 nonzeros
Found heuristic solution: objective 30.0000000
Variable types: 0 continuous, 225 integer (225 binary)

Root relaxation: cutoff, 292 iterations, 0.17 seconds (0.00 work units)

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

     0     0     cutoff    0        30.00000   30.00000  0.00%     -    1s

Explored 1 nodes (292 simplex iterations) in 1.65 seconds (0.00 work units)
Thread count was 4 (of 4 available processors)

Solution count 2: 30 -0
```

Figura 1: Mensaje arrojado luego de correr el programa creado

Finalmente, se observa que las tres partes que componen esta pregunta se conectan en el último ítem ya que lo que busca dicha parte es aumentar el compromiso, es decir que aumenta la cantidad de roles asumidos por reunión, de forma que el tiempo que aporta cada rol a la reunión menos el que gasta cada persona en asumirlo sea el máximo. Entre más personas asuman un rol, donde dicho rol tiene un valor positivo de "ganancia de tiempo" (T-t_i sea positivo), mayor será la eficiencia de la reunión.

3. Problema 2: PRVP

En el archivo *.ipynb* adjunto con esta sección del informe hay 4 cajas de código principales. La primera tiene el propósito de leer los datos del Excel 'Pacientes' y dejarlos de forma cómoda para trabajar.

El segundo es el programa del primer caso, donde solamente hay un funcionario disponible para trabajar. Este se modela con el nombre 'PRVP'.

El tercer bloque programa al caso en el que hay M funcionarios disponibles cada día para trabajar, es el caso general del anterior. Este se modela con el nombre de 'PRVPconM', este se cumple $\forall$ M (incluyendo al 1).

El cuarto es aquel que contiene el programa necesario para graficar el primer caso.

3.1. Variables de Decisión

3.1.1. V.D. caso M=1

- X_{ki} = 1 si para un día k $\in$ K se visita el lugar i $\in$ N, 0 si no.
- Y_{kij} = 1 si para un día k $\in$ K, se realiza el viaje desde i a j con i,j $\in$ N, 0 si no.
- U_{kj} es una variable de decisión continua que puede tomar valores entre 1-N (en este caso entre 1-185), y representa el orden en el que se visita al paciente j $\in$ N en un día k $\in$ K.

3.1.2. V.D. caso $\forall$ M

- X_{mki} = 1 si para un día k $\in$ K el funcionario m $\in$ M visita al paciente i $\in$ N.
- Y_{mkij} = 1 si para un día k $\in$ K, el funcionario m $\in$ M realiza el tramo que va desde el paciente i hasta el paciente j con i,j $\in$ N.
- U_{mkj} es una variable de decisión continua que puede tomar valores entre 1-N (en este caso entre 1-185), y representa el orden en el que el funcionario m $\in$ M visita al paciente j $\in$N un día k $\in$K.

Con N= 0....N, donde N= N_e + N_t, es decir es el conjunto con la cantidad total de lugares a visitar. Donde 0 representa al centro médico, los siguientes N_t (en este caso 1-40) representa las ubicaciones de los pacientes con tratamientos en orden de primero a ultimo ordenados en *Excel*. Los siguientes N_e (en este caso 41-185) son las ubicaciones de los pacientes con citas ordenados en orden de primero a ultimo en el *Excel*.

K= 1...5, los días de la semana. Para el programa y este informe se usara *auxN* para referirse al conjunto N $\setminus\{0\}$.

3.2. Restricciones

Las restricciones en ambos casos M = 1 y ∀ M son las mismas, pero se modifican levemente, por ejemplo algunas además de cumplirse para todo k ∈ K, se deben cumplir para todo funcionario m ∈ M. Para entender mejor las diferencias, en cada restricción estará el caso M=1 y el caso M>1.

Además se crearon 3 categorías que representan los tipos de restricciones.

3.2.1. Relaciones entre las Variables

1. Y_{kij} $\Longrightarrow$X$_{ki}$:

- Caso M=1: "Para un día k fijo, si es que existe el tramo de i a j, entonces el día k se visita al paciente i"

$$x_{ki} \leq Cs \sum_{i \in N} y_{kij} \tag{4}$$

Donde C_s es un numero mayor a 1 arbitrario. C_i es un numero menor a 1.

$$Ci \sum_{i \in N} y_{kij} \leq x_{ki} \tag{5}$$

- Caso ∀ M: "Para un día k y funcionario m fijos, si es que existe el tramo de i a j, entonces el funcionario m visita al paciente i el día k"

$$x_{mki} \leq Cs \sum_{i \in N} y_{mkij} \tag{6}$$

$$Ci \sum_{i \in N} y_{mkij} \leq x_{mki} \tag{7}$$

Donde C_s es un numero mayor a 1 arbitrario. C_i es un numero menor a 1.

3.2.2. Cumplimiento de Citas y Tratamientos

f_i representa la frecuencia de visita que requiere cada paciente i ∈ *auxN*. d_i representa el día agendado por cada paciente i ∈ Ne.

1. "Se deben visitar a los pacientes en la frecuencia semanal que corresponde"

- Caso M=1: Para un paciente i ∈ *auxN* fijo.

$$\sum_{k \in K} x_{ki} * k = f_i \tag{8}$$

- Caso ∀ M: Para un paciente i ∈ *auxN* fijo.

$$\sum_{m \in M} \sum_{k \in K} x_{mki} == f_i \tag{9}$$

2. "Los pacientes con citas se les debe visitar el día que corresponde"

   - Caso M=1: Para un paciente i $\in$ Ne fijo.

$$\sum_{k\in K} x_{ki} * k = d_i \tag{10}$$

   - Caso $\forall$ M: Para un paciente i $\in$ Ne fijo.

$$\sum_{m\in M} \sum_{k\in K} x_{mki} * k = d_i \tag{11}$$

3.2.3. Restricciones de Viaje

1. "Puede salir a lo más un vehículo desde la casa de un paciente"

   - Caso M=1: Para un día k y un paciente i $\in$ N fijos. La suma total de viajes saliendo desde i es menor o igual a 1.

$$\sum_{j\in N} y_{kij} \leq 1 \tag{12}$$

   - Caso $\forall$ M: Para un día k y paciente i $\in$ $auxN$ fijos. La suma total de viajes realizados desde i es menor o igual a 1.

$$\sum_{j\in N} \sum_{m\in M} y_{mkij} \leq 1 \tag{13}$$

2. "La misma cantidad de salidas que llegadas"

   - Caso M=1: Para un día k $\in$ K un funcionario m $\in$ M y un lugar i $\in$ N fijos.

$$\sum_{j\in N} y_{kij} == \sum_{l\in N} y_{kli} \tag{14}$$

   - Caso $\forall$ M: Para un día k $\in$ K un funcionario m $\in$ M y un lugar i $\in$ N fijos.

$$\sum_{j\in N} y_{mkij} == \sum_{l\in N} y_{mkli} \tag{15}$$

   Notemos que con esta condición aseguramos que se arme un loop, a la vez aseguramos que la cantidad de salidas desde el hospital sea igual a las llegadas. Esto es especialmente importante porque, por la naturaleza del problema no podemos imponer que los 6 trabajadores salgan a trabajar, pues existe la posibilidad de que algún día convenga que salgan menos funcionarios. Con esta condición nos aseguramos que independiente de lo que convenga todas las rutas sean loops que vuelven al C.M.

3. "No hay viajes que salgan desde la residencia del paciente i y lleguen al mismo"

- Caso M=1: Para un día k ∈ K y un paciente i ∈ *auxN* fijos.

$$y_{kii} == 0 \tag{16}$$

- Caso ∀ M: Para un día k ∈ K, un funcionario m ∈ M y un paciente i ∈ *auxN* fijos.

$$y_{mkii} == 0 \tag{17}$$

4. "Los viajes no son reversibles"

- Caso M=1: Para un día k ∈ K, lugares i,j ∈ N fijos:

$$y_{kij} * y_{kji} == 0 \tag{18}$$

- Caso ∀ M: Para un día k ∈ K, un funcionario m ∈ M lugares i,j ∈ N fijos:

$$y_{mkij} * y_{mkji} == 0 \tag{19}$$

5. "Salidas desde el Centro Médico.En las siguientes condiciones imponemos que debe haber exactamente x cantidad de salidas desde el centro médico. Esto no es tan así pues quizás algún día conviene que solamente salga la mitad de funcionarios o que quizás no salga nadie. Existe la posibilidad de que el viaje sea desde i=0 a j=0 y eso es análogo a que no hayan viajes.

- Caso M=1: Para k ∈ K fijo debe haber exactamente 1 salida desde el Centro Médico.

$$\sum_{j \in N} y_{k0j} == 1 \tag{20}$$

- Caso ∀ M: Para k ∈ K y un funcionario m ∈ M fijo debe haber exactamente una salida desde el Centro Médico.

$$\sum_{j \in N} y_{mk0j} == 1 \tag{21}$$

6. "No hay sub-tours, estrategia mtz"

- Caso M=1: Para un día k, un paciente de partida i ∈ N{0} y un paciente de llegada j ∈N{0} fijos.

$$u_{ki} + 1 \leq u_{kj} + L * (1 - y_{kij}) \tag{22}$$

- Caso ∀ M: Para un día k, funcionario m, un paciente de partida i ∈ $N\{0\}$ y un paciente de llegada j ∈ N{0} fijos.

$$u_{mki} + 1 \leq u_{mkj} + L * (1 - y_{mkij}) \tag{23}$$

En el caso general de mtz se toma L como l cantidad todal de nodos visitados ese día, pero también funciona con un numero más grande, para este programa tomamos $L = 185$.

3.3. Función Objetivo

Minimizar el tiempo total. Acá se usa la variable T_{ij} que representa el tiempo que se demora en hacer el tramo desde i a j.

3.3.1. Caso M=1

$$min \sum_{j\in N}\sum_{i\in N}\sum_{k\in K} T_{ij} * y_{kij} + S + P \tag{24}$$

3.3.2. Caso ∀M

$$min \sum_{m\in M}\sum_{j\in N}\sum_{i\in N}\sum_{k\in K} T_{ij} * y_{mkij} + S + P \tag{25}$$

Con S y P definidos como el tiempo total en atender a los pacientes. De forma general queda:

$$P = 20 * \sum_{m\in M}\sum_{k\in K}\sum_{i\in Nt} x_{mki} \tag{26}$$

$$S = 10 * \sum_{m\in M}\sum_{k\in K}\sum_{i\in Ne} x_{mki} \tag{27}$$

3.4. Resultados

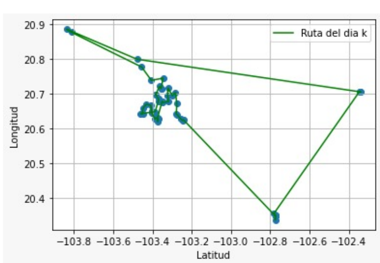
3.4.1. Caso M=1

Según nuestros resultados de la programación de este problema en gurobi, el mínimo de tiempo total semanal queda en $92,623$ horas.

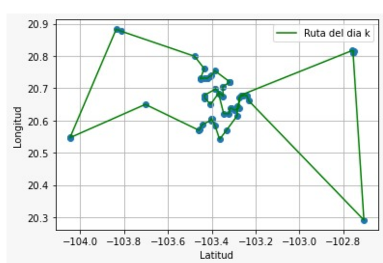
```
Solution count 10: 5557.39 5574.48 5586.6 ... 5641.54

Time limit reached
Best objective 5.557388263135e+03, best bound 4.966792767397e+03, gap 10.
6272%
El tiempo minimo es 5557.388263135216 minutos totales esa semana
```

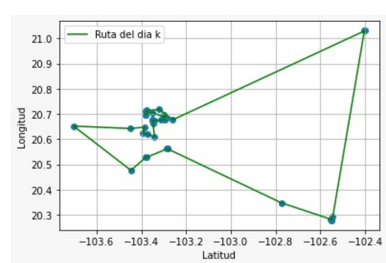
Figura 2: Mensaje arrojado luego de correr el modelo 'PRVP'



(a) Lunes

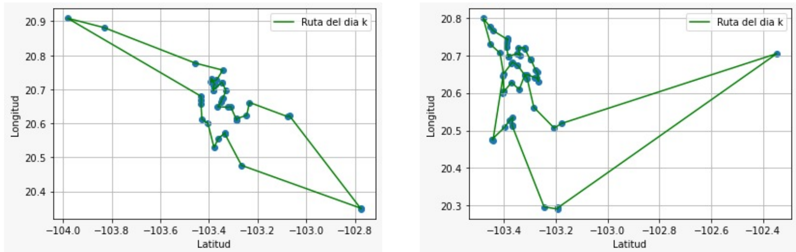


(b) Martes



(c) Miércoles

Figura 3: Resultados de los gráficos para cada día



(a) Jueves (b) Viernes

Figura 4: Resultados de los gráficos para cada día

3.4.2. Caso M=6

```
Explored 100592 nodes (6585772 simplex iterations) in 3830.12 seconds (2217.56 work units)
Thread count was 8 (of 8 available processors)

Solution count 0

Solve interrupted
Best objective -, best bound 4.859642599976e+03, gap -
El tiempo minimo es inf minutos totales esa semana
```

Figura 5: Mensaje arrojado luego de correr el programa creado

3.4.3. Análisis de resultados

El planteamiento para el caso general donde M toma cualquier valor entero positivo es el mismo que para el caso de M=1, solo que para ciertas restricciones se suma en m $\in$ M, o se deja para m fijo. Lo ideal habría sido agregarle restricciones nuevas o cambiar solamente un par, pero el problema es que mtz no se podría adaptar a ese caso.

Para resolver el caso general puede haber ocurrido que faltó definir alguna restricción, o bien es muy caro en términos de nodos recorridos

3.5. BONUS

El hecho de que se pueda reservar una hora dentro de un rango de horario producirá que se deba agregar una restricción tal que: "la suma de los tiempos acumulados de los pacientes que atendieron antes que el paciente j"(que reservo una hora R_{jk}) debe cumplir lo siguiente:

Para k $\in$ K fijo, i,j $\in$ rut:

$$u_{ij} \leq R_{jk} - t_{ij} \tag{28}$$

Esto producirá que para cada R_{jk} distinto de 0 (ósea que existe una reserva de hora del paciente j en el día k) se alcanzará a llegar a la hora de atención para el paciente j.

4. Conclusión

Modelar es una herramienta muy útil en este mundo lleno de datos y problemáticas a resolver. Saber traducir una situación o problema que aqueja a una empresa/institución a variables matemáticas, que se rigen bajo determinadas restricciones, para obtener una solución óptima aproximada, es un trabajo arduo y complejo.

Este informe plasma completamente lo descrito anteriormente, modelando 2 problemas distintos en hoja y lápiz, para luego ser optimizados mediante *Phyton* y el programa *Gurobi*. Estos últimos tomaron un rol significativo a la hora de resolverlos. De no ser por ellos, por la cantidad de iteraciones y de ecuaciones a relacionar, estos problemas no podrían haber sido resueltos.

Por un lado, el problema 1, el cual tiene similitud con el famoso problema de "La mochila", requiere de saber elegir estratégicamente las variables de decisión de forma que las restricciones sean obvias al momento de plantearlas.

Lo mismo pasa para el problema 2, el cual es una variación del famoso problema del "Vendedor viajero". Saber elegir las variables de decisión de manera correcta hace que el problema sea mucho más llevadero y que las restricciones salgan de manera simple. Es más, los problemas tienen una relación secuencial, es decir, cada parte se relacionan entre sí, produciendo que en la primera parte de cada problema, las variables de decisión que se tomen, continuarán siendo válidas para el resto de la pregunta, solo que a medida que esta avanza se le agregarán cosas más específicas. Para este problema fue crucial comenzar a programar lo antes posible, e implementar alguna función de prueba que permita visualizar la ruta. El desafío principal fue plantear la restricción que evitara los "subtours", en especial para un caso donde hay varios funcionarios disponibles para trabajar.

Finalmente es importante agregar que para las futuras tareas es necesario tener una base de conocimientos de programación bastante sólida, ya que de no ser así el proceso de traspasar de la hoja escrita a mano a un algoritmo de programación será más lento.