Problema de optimización

Trabajo de Curso

Universidad de Sevilla
Ingeniería Informática Tecnologías Informáticas
Matemática aplicada a Sistemas de Información - Tercer curso

Juan Arteaga Carmona
Herrera, Sevilla, España
JuanArteaga@andalu30.me
Participación: 50 %

Enrique Ramos Miró
Sanlúcar la Mayor, Sevilla, España
kikeramos9@gmail.com
Participación: 50 %

25 de mayo de 2019

Índice

Índice de tablas

1. Descripción del problema general de asignación

A continuación se muestra el ejercicio 15 del documento "Supplementary Exercises", el cual ha sido asignado a nuestro grupo.

1.1. Problema asignado al grupo

After qualifying, medical students must take two six-month jobs in hospital departments, but they cannot take both jobs in the same department. A hospital has four students and vacancies in four departments: Casualty, Maternity, medical and Surgical. The number of fatal mistakes each student will make in each department is given by the table 1.1.

	Casualty	Maternity	Medical	Surgical
Student 1	3	0	2	6
Student 2	2	1	4	5
Student 3	4	2	5	7
Student 4	2	0	2	4

1. How should they be allotted to departments for the first job so as to minimize the total mistakes?

2. Given that allocation, how should they be allotted for the second six months so no one stays in the same job, and mistakes are minimized?

1.2. Problema general de asignación

Un problema de asignación consiste en encontrar un emparejamiento de peso óptimo en un grafo bipartito ponderado, donde los pesos de las aristas se corresponden con los costes de las asignaciones, valor que se debe de minimizar. [4]. En nuestro caso nos encontramos con estudiantes que tienen que ser asignados a unos departamentos.

Generalmente, en los problemas de asignación a cada recurso se le asigna una única tarea, sin embargo, es posible interpretar nuestro problema de tal forma que no sea estrictamente necesario que un departamento tenga una única vacante. En este caso es posible que más de un estudiante acabe en el mismo departamento ya que han sido asignados al departamento en el que tendrán menos errores, algo más que razonable en la vida real. Por esto mismo hemos tenido en cuenta esta interpretación y también le daremos una solución.

2. Métodos de resolución

2.1. El método húngaro

El método húngaro [2] es un método de optimización de problemas de asignación.

Este método utiliza la propiedad de reducción de matrices para reducir la matriz original de costes, hasta que los costes $C_{i,j}$ asociados con la asignación óptima, sean cero y todos los otros costes sean no negativos. En cada iteración del método húngaro, se reduce la matriz de tal manera que haya al menos un cero en cada fila y columna.

El algoritmo está diseñado para la resolución de problemas de minimización únicamente.

Procedimiento:
El Método Húngaro consta de los siguientes pasos:

- **Paso 1:** En la matriz original de costo, identificar el mínimo de cada fila y restarlo de todos los elementos de la fila.
- **Paso 2:** En la matriz que resulte del Paso 1, identificar el mínimo de cada columna, y restarlo de todos los elementos de la columna.
- **Paso 3:** Identificar la solución óptima como la asignación factible asociada con los elementos cero de la matriz obtenida en el Paso 2.

2.2. Resolución mediante SAGE

SageMath [3], comúnmente conocido por su antiguo nombre, SAGE, es un software open-source matemático creado sobre diversas librerias del lenguaje de programación Python [1].

Para resolver problemas de programacion lineal utilizaremos la clase `MixedIntegerLinearProgram`, que nos permite definir este tipo de problemas.
Definiremos las variables que aparecen en el problema con
`x=p.new_variable(nonnegative=True,binary=true)`, donde es importante poner a `true` el parámetro `binary`, declararemos el objetivo con `set_objective` y finalmente resolveremos el problema con el método `solve`
La solución al ejercicio propuesto con SAGE, teniendo en cuenta las dos interpretaciones, se puede ver en el Anexo 1.

3. Resolución del problema propuesto

3.1. Resolución del problema considerando que no hay un mínimo de vacantes en cada departamento

3.1.1. Modelado mediante Programación lineal del problema propuesto

Tal y como hemos visto en el apartado 1.1, los datos de nuestro problema de asignación se nos han dado en formato tabla. De esta tabla, los datos que más nos interesan son los errores que van a cometer los alumnos, que llamaremos c.
Para modelar el problema mediante programación lineal nos ayudaremos de 16 variables binarias (que llamaremos $x_{i,j}$ con $i, j = 1, .., 4$) que indicarán en que departamento serán asignados los alumnos.

Nuestra ecuación a minimizar será $\sum c_i x_{i,j}$, con $i, j = 1, .., 4$
Además, necesitaremos varias restricciones para asegurarnos de obtener una solución correcta. En este caso necesitaremos de 5 restricciones, 4 para indicar que cada alumno necesita ser incluido a un único departamento y otra más para indicar que las variables son binarias.

De esta forma, nos quedaría el siguiente problema:

Sea $Z(x) = 3x_{1,1} + 0x_{1,2} + 2x_{1,3} + 6x_{1,4} + 2x_{2,1} + x_{2,2} + 4x_{2,3} + 5x_{2,4} + 4x_{3,1} + 2x_{3,2} + 5x_{3,3} + 7x_{3,4} + 2x_{4,1} + 0x_{4,2} + 2x_{4,3} + 4x_{4,4}$

$$\begin{aligned} \text{minimize} \quad & Z(x) \\ \text{subject to} \quad & x_{1,1} + x_{1,2} + x_{1,3} + x_{1,4} = 1, \\ & x_{2,1} + x_{2,2} + x_{2,3} + x_{2,4} = 1, \\ & x_{3,1} + x_{3,2} + x_{3,3} + x_{3,4} = 1, \\ & x_{4,1} + x_{4,2} + x_{4,3} + x_{4,4} = 1, \\ & \qquad\qquad x_{i,j} \in \{0, 1\} \; i, j = 1, .., 4 \end{aligned}$$

3.1.2. Aplicación de un método de resolución para obtener el resultado al primer apartado

Al resolver este problema de programación lineal e interpretar los resultados nos encontraremos con la solución al primer apartado del problema.

Un ejemplo de resolución se puede encontrar en el anexo 1, seccion 1.2, en el que utilizamos SAGE para resolver este problema.

3.1.3. Aplicación de un método de resolución para obtener el resultado al segundo apartado

Para solucionar el segundo apartado necesitaremos tener en cuenta los resultados del primero ya que como vimos en el apartado 1.1 ningún alumno puede repetir departamento. Para asegurarnos de que esto es así realizaremos algunos cambios a los costes de nuestro problema, intercambiando el valor de los costes correspondientes con las soluciones del primer apartado por M, un número lo suficientemente grande para que esta solución sea penalizada y, de esta forma, asegurarnos de que los alumnos no pueden repetir departamento.

La solución mediante sage se puede observar en el anexo 1, seccion 1.3.

3.2. Resolución del problema considerando que sólo hay una vacante en cada departamento

3.2.1. Modelado mediante Programación lineal del problema propuesto

El problema que nos resulta al considerar este caso es prácticamente el mismo que hemos visto en el apartado 3.1.1. Sin embargo existe una diferencia muy importante, debemos introducir varias restricciones que nos indiquen que solo existe una vacante en cada departamento.

De esta forma, al igual que en el caso anterior, para modelar el problema mediante programación lineal nos ayudaremos de 16 variables binarias (que llamaremos $x_{i,j}$, con $i, j = 1, .., 4$) que indicarán en que departamento serán asignados los alumnos.

Nuestra ecuación a minimizar será $\sum c_i x_{i,j}$ (con $i, j = 1, .., 4$) donde c serán los errores que pueden cometer cada alumno en cada departamento y, además, tendremos las restricciones anteriores y 4 más, que nos indican que todos los departamentos reciben un alumno.

De esta forma, nos quedaría el siguiente problema:

Sea $Z(x) = 3x_{1,1} + 0x_{1,2} + 2x_{1,3} + 6x_{1,4} + 2x_{2,1} + x_{2,2} + 4x_{2,3} + 5x_{2,4} + 4x_{3,1} + 2x_{3,2} + 5x_{3,3} + 7x_{3,4} + 2x_{4,1} + 0x_{4,2} + 2x_{4,3} + 4x_{4,4}$

$$
\begin{aligned}
\text{minimize} \quad & Z(x) \\
\text{subject to} \quad & x_{1,1} + x_{1,2} + x_{1,3} + x_{1,4} = 1, \\
& x_{2,1} + x_{2,2} + x_{2,3} + x_{2,4} = 1, \\
& x_{3,1} + x_{3,2} + x_{3,3} + x_{3,4} = 1, \\
& x_{4,1} + x_{4,2} + x_{4,3} + x_{4,4} = 1, \\
& x_{1,1} + x_{2,1} + x_{3,1} + x_{4,1} = 1, \\
& x_{1,2} + x_{2,2} + x_{3,2} + x_{4,2} = 1, \\
& x_{1,3} + x_{2,3} + x_{3,3} + x_{4,3} = 1, \\
& x_{1,4} + x_{2,4} + x_{3,4} + x_{4,4} = 1, \\
& \qquad\qquad\qquad x_{i,j} \in \{0,1\} \; i,j = 1,..,4
\end{aligned}
$$

3.2.2. Aplicación de un método de resolución para obtener el resultado al primer apartado

Al resolver este problema de programación lineal e interpretar los resultados nos encontraremos con la solución al primer apartado del problema.

En esta ocasión es posible resolver el problema mediante 2 métodos distintos, mediante SAGE, al igual que en el apartado 3.1.2 y con el método húngaro que se ha explicado en el apartado 2.1.

- La resolución mediante SAGE se puede ver en el anexo 1.
- La resolución mediante el método húngaro se encuentra a continuación.

	Casualty	Maternity	Medical	Surgical	
Student 1	3	0	2	6	0
Student 2	2	1	4	5	1
Student 3	4	2	5	7	2
Student 4	2	0	2	4	0

Tabla 1: Paso 1 del método húngaro, primer apartado

En primer lugar, en el paso 1 del método húngaro vemos cuál es el número menor de cada fila, en este caso 0,1,2 y 0 para la fila 1,2,3 y 4 respectivamente. (Escrito a la derecha de la tabla).

Y a continuación se va a restar cada número por su fila (Nos damos cuenta que si es 0, se queda la fila tal cuál), es decir, por ejemplo para la fila 2 sería: 2-1 = 1, 1-1 = 0, 4-1 = 3 y 5-1 = 4. Y así sucesivamente por cada fila, donde nos quedaría finalmente la tabla 2 (justo debajo).

	Casualty	Maternity	Medical	Surgical
Student 1	3	0	2	6
Student 2	1	0	3	4
Student 3	3	0	3	5
Student 4	2	0	2	4
	1		2	4

Tabla 2: Paso 2 del método húngaro, primer apartado

En este paso 2 del método, hacemos lo mismo que antes en el paso 1 pero con las columnas, es decir, buscamos el número mínimo de cada columna (nos damos cuenta que si hay un 0 nos ahorramos el ponerlo porque nos quedaría igual) y se lo restamos a dicha columna. Nos queda la tabla 3 (igualmente, la que está justo debajo).

	Casualty	Maternity	Medical	Surgical
Student 1	~~2~~	~~0~~	~~0~~	~~2~~
Student 2	~~0~~	~~0~~	~~1~~	~~0~~
Student 3	1	~~0~~	1	1
Student 4	~~1~~	~~0~~	~~0~~	~~0~~

Tabla 3: Paso 3 del método húngaro, primer apartado

En este paso 3, donde nos encontremos un 0, vamos a hacer una línea para 'eliminarlos'. Nos salen 4 líneas (3 horizontales y 1 vertical) y como nuestra matriz es 4x4, es decir, tenemos 4 estudiantes para 4 puestos de trabajo, habríamos acabado.
Donde se ubican los ceros son los lugares posibles a donde podemos enviar a cada estudiante con su correspondiente departamento.

- **Student 1:** Maternity, Medical
- **Student 2:** Casualty, Maternity, Surgical
- **Student 3:** Maternity
- **Student 4:** Maternity, Medical, Surgical

Cómo vemos hemos asociado a cada estudiante con los ceros de la tabla y su correspondiente departamento, pues bien, al estudiante 3 ya le podríamos asignar Maternity porque sólo tiene ese 0 en esa fila, luego ya a ningún otro más se le podría asignar ese departamento así que tacharíamos Maternity de los demás estudiantes.

Y vemos que si le tachamos Maternity al estudiante 1 sólo quedaría Medical, así que le asignamos Medical al estudiante 1 e igualmente lo volveríamos a tachar si le apareciera a otro estudiante. El estudiante 4 le asignamos Surgical, el cuál sería el único, porque tanto Maternity como Medical estarían tachados. Y por último al estudiante 2 le asignamos Casualty.

Quedaría tal que así:

- **Student1:** Medical
- **Student2:** Casualty
- **Student3:** Maternity
- **Student4:** Surgical

Por último nos vamos a la primera tabla, 'Tabla 1' y vemos los valores de la tabla asociados a estudiante 1 y Medical, al estudiante 2 y Casualty, al estudiante 3 y Maternity y al estudiante 4 y Surgical que serían 2, 2, 2 y 4 respectivamente. Los sumamos y nos da el costo mínimo final. $(2 + 2 + 2 + 4 = 10)$

3.2.3. Aplicación de un método de resolución para obtener el resultado al segundo apartado

Para solucionar el segundo apartado necesitaremos tener en cuenta los resultados del primero ya que como vimos en el apartado 1.1 ningún alumno puede repetir departamento.

Una vez mas, realizaremos algunos cambios a los costes de nuestro problema, intercambiando el valor de los costes correspondientes con las soluciones del primer apartado por M, un número lo suficientemente grande para que esta solución sea penalizada.

Al igual que para el primer apartado, es posible solucionarlo mediante dos métodos distintos, SAGE y el método húngaro.

- La resolución mediante SAGE se puede ver, una vez mas, en el anexo 1.
- La resolución mediante el método húngaro se encuentra a continuación.

	Casualty	Maternity	Medical	Surgical	
Student 1	3	0	M	6	0
Student 2	M	1	4	5	1
Student 3	4	M	5	7	4
Student 4	2	0	2	M	0

Tabla 4: Paso 1 del método húngaro, segundo apartado

Para el segundo apartado lo que hacemos es como la resolución mediante SAGE, y es que le metemos una variable, en este caso la llamamos M, suficientemente grande donde antes nos había salido las soluciones para que penalice mucho y no vuelvan a repetirse (quedaría como la Tabla 4), por lo demás el método de resolución es el mismo que en el apartado 1, eligiendo en primer lugar el número más pequeño de cada fila y restándolo.

	Casualty	Maternity	Medical	Surgical
Student 1	3	0	M	6
Student 2	M-1	0	3	4
Student 3	0	M-4	1	3
Student 4	2	0	2	M
			1	3

Tabla 5: Paso 2 del método húngaro, segundo apartado

Al igual que en el Paso 2 del apartado anterior ahora en vez el número mas pequeño de cada fila se cogería el número más pequeño de cada columna y se le restaría. Obviando como antes si hubiera ceros porque al restarlo se quedarían igual tanto filas como columnas.

	Casualty	Maternity	Medical	Surgical
Student 1	3	0	M-1	3
Student 2	M-1	0	2	1
Student 3	~~0~~	~~M-4~~	~~0~~	~~0~~
Student 4	2	0	1	M-3

Tabla 6: Paso 3 del método húngaro, segundo apartado

En este paso 3 tenemos una novedad y es que al tachar con una línea, una vez ya finalizado el paso 2 como en el apartado 1 que teníamos que 'eliminar' los ceros, nos sale solamente 2 líneas y como la matriz es 4x4 como dijimos anteriormente nos debe de salir 4 líneas para acabar, luego hacemos lo siguiente:
cogemos el menor número de los que no están tachado, en nuestro caso es el 1 (Recuerdo: M >0 sufic. grande), como hay dos no pasa nada, cogemos uno (está señalado con el círculo) y restamos todos los demás que no están tachados por 1 inclusive el que hemos señalado.
Por ejemplo tal que así, $3-1=2$, $M-1(-1)=M-2$, $2-1=1$, $M-1(-1)=M-2$, $2-1=1$, $1-1=0$, $3-1=1$, $1-1=0$, $M-3(-1)=M-4$. Y el número donde se cortan ambas líneas, es decir, en nuestro caso el $M-4$, se le suma un 1, $M-4+1=M-3$. Después de esto nos quedaría la Tabla 7.

	Casualty	Maternity	Medical	Surgical
Student 1	2	0	M-2	2
Student 2	$M-2$	0	1	0
Student 3	0	M-3	0	0
Student 4	1	0	0	$M-4$

Tabla 7: Paso 4 del método húngaro, segundo apartado

En este paso con la Tabla 7 tal cuál está, volvemos a tachar las líneas o columnas donde aparezcan ceros para 'eliminarlos' como hemos estado haciendo hasta ahora.
Como vemos en la tabla ahora nos sale 4 líneas, por lo que habríamos acabado nuestro método y como hicimos antes en el apartado anterior que decíamos que donde se ubiquen los ceros son los lugares posibles a donde podemos enviar a cada estudiante con su correspondiente departamento. Nos quedaría lo siguiente.

- **Student 1:** Maternity
- **Student 2:** Maternity, Surgical
- **Student 3:** Casualty, Medical, Surgical
- **Student 4:** Maternity, Medical

Por el mismo procedimiento que antes le asignamos al estudiante 1 Maternity, al 2 Surgical al 3 Casualty y al 4 Medical.
Finalmente, la solución al ejercicio seria la siguiente:

- **Student 1:** Maternity
- **Student 2:** Surgical
- **Student 3:** Casualty
- **Student 4:** Medical

Con los valores asociados a cada uno 0,5,4 y 2, luego su suma sería $0+5+4+2 = 11$

Referencias

[1] Python Software Foundation. Welcome to python, 2019.

[2] inves.op.ittol. El método húngaro, 2016.

[3] SageMath Project. Sagemath - open-sorce mathematics software, 2019.

[4] Wikipedia. Problema de la asignación, 2016.

Anexo

A continuación se incluye una copia en PDF del archivo de SAGE en el que se resuelve el problema teniendo en cuenta las dos consideraciones.

Anexo 1: Resolución de los problemas con SAGE

Problema de optimización

Trabajo de Curso

Universidad de Sevilla

Ingeniería Informática Tecnologías Informáticas

Matemática aplicada a Sistemas de Información - Tercer curso

Juan Arteaga Carmona
Herrera, Sevilla, España
JuanArteaga@andalu30.me
Participación: 50%

Enrique Ramos Miró
Sanlúcar la Mayor, Sevilla, España
kikeramos9@gmail.com
Participación: 50%

1. Resolución del problema considerando que no hay un mínimo de vacantes en cada departamento.

1.1 Modelado mediante Programación lineal del problema propuesto.

```
p=MixedIntegerLinearProgram(maximization=False)
x=p.new_variable(nonnegative=True,binary=true)
```

```
p.add_constraint(x[11]+x[12]+x[13]+x[14]==1)
p.add_constraint(x[21]+x[22]+x[23]+x[24]==1)
p.add_constraint(x[31]+x[32]+x[33]+x[34]==1)
p.add_constraint(x[41]+x[42]+x[43]+x[44]==1)
```

```
p.set_objective(3*x[11]+0*x[12]+2*x[13]+6*x[14]+2*x[21]+x[22]+4*x[23]+5*x[24]+4*x[31]+2*x[32]+5*x[33]+7*x[34]+2*x[41]+0*x[42]+2*x[43]+4*x[44])
```

1.2 Aplicación de un metodo de resolución para obtener el resultado al primer apartado

```
p.solve()
```

```
3.0
```

```
p.get_values(x)
```

```
{11: 0.0,
 12: 1.0,
 13: 0.0,
 14: 0.0,
 21: 0.0,
 22: 1.0,
 23: 0.0,
 24: 0.0,
 31: 0.0,
 32: 1.0,
 33: 0.0,
 34: 0.0,
 41: 0.0,
 42: 1.0,
 43: 0.0,
 44: 0.0}
```

Como podemos ver en el resultado anterior las variables seleccionadas han sido 12, 22, 32, 42, lo que indica que todos los estudiantes han sido seleccionado en el departamento de maternidad ya que esta es la mejor forma de minimizar los errores que se producirán.

1.3 Aplicación de un metodo de resolución para obtener el resultado al segundo apartado

```
p=MixedIntegerLinearProgram(maximization=False)
```

```
x=p.new_variable(nonnegative=True,binary=true)
```

```
p.add_constraint(x[11]+x[12]+x[13]+x[14]==1)
p.add_constraint(x[21]+x[22]+x[23]+x[24]==1)
p.add_constraint(x[31]+x[32]+x[33]+x[34]==1)
p.add_constraint(x[41]+x[42]+x[43]+x[44]==1)
```

```
bigM = 10000000000
```

```
p.set_objective(
3*x[11]+bigM*x[12]+2*x[13]+6*x[14]+
2*x[21]+bigM*x[22]+4*x[23]+5*x[24]+
4*x[31]+bigM*x[32]+5*x[33]+7*x[34]+
2*x[41]+bigM*x[42]+2*x[43]+4*x[44])
```

```
p.solve()
```

```
10.0
```

```
p.get_values(x)
```

```
{11: 0.0,
 12: 0.0,
```

```
 13: 1.0,
 14: 0.0,
 21: 1.0,
 22: 0.0,
 23: 0.0,
 24: 0.0,
 31: 1.0,
 32: 0.0,
 33: 0.0,
 34: 0.0,
 41: 0.0,
 42: 0.0,
 43: 1.0,
 44: 0.0}
```

En este caso obtenemos un resultado más variado:

- El estudiante 1 ha sido asignado al departamento 3 (Medicina)
- El estudiante 2 ha sido asignado al departamento 1 (Urgencias)
- El estudiante 3 ha sido asignado al departamento 1 (Urgencias)
- El estudiante 4 puede ser asignado tanto al departamento 1 como al 3 (Urgencias y Medicina), ya que tendrían el mismo número de errores fatales (2) en ambos departamentos. De modo que cada vez que ejecutemos la función 'p.get_values(x)' unas veces saldrá que se le ha sido asignado el departamento 1 y otras el 3

2. Resolución del problema considerando que solo hay una vacante en cada departamento.

La solución será la misma que se optiene por el metodo húngaro.

2.1 Modelado mediante Programación lineal del problema propuesto.

```
p=MixedIntegerLinearProgram(maximization=False)
x=p.new_variable(nonnegative=True,binary=true)
```

```
# Todos los alumnos tienen que ser destinados a un departamento
p.add_constraint(x[11]+x[12]+x[13]+x[14]==1)
p.add_constraint(x[21]+x[22]+x[23]+x[24]==1)
p.add_constraint(x[31]+x[32]+x[33]+x[34]==1)
p.add_constraint(x[41]+x[42]+x[43]+x[44]==1)

# Todos los departamentos tienen que recibir algún alumno
p.add_constraint(x[11]+x[21]+x[31]+x[41]==1)
p.add_constraint(x[12]+x[22]+x[32]+x[42]==1)
p.add_constraint(x[13]+x[23]+x[33]+x[43]==1)
p.add_constraint(x[14]+x[24]+x[34]+x[44]==1)
```

```
p.set_objective(3*x[11]+0*x[12]+2*x[13]+6*x[14]+2*x[21]+x[22]+4*x[23]+5*x[24]+4*x[31]+2*x[32]+5*x[33]+7*x[34]+2*x[41]+0*x[42]+2*x[43]+4*x[44])
```

2.2 Aplicación de un metodo de resolución para obtener el resultado al primer apartado

```
p.solve()
```

```
10.0
```

```
p.get_values(x)
```

```
{11: 0.0,
 12: 0.0,
 13: 1.0,
 14: 0.0,
 21: 1.0,
 22: 0.0,
 23: 0.0,
 24: 0.0,
 31: 0.0,
 32: 1.0,
 33: 0.0,
 34: 0.0,
 41: 0.0,
 42: 0.0,
 43: 0.0,
 44: 1.0}
```

Como podemos observar, en la solucion:

- El alumno 1 será destinado al departamento 3 (Medico)
- El alumno 2 será destinado al departamento 1 (Urgencias)
- El alumno 3 será destinado al departamento 2 (Maternidad)
- El alumno 4 será destinado al departamento 4 (Cirugía)

2.3 Aplicación de un metodo de resolución para obtener el resultado al segundo apartado

```
p=MixedIntegerLinearProgram(maximization=False)
x=p.new_variable(nonnegative=True,binary=true)
```

```
# Todos los alumnos tienen que ser destinados a un departamento
p.add_constraint(x[11]+x[12]+x[13]+x[14]==1)
p.add_constraint(x[21]+x[22]+x[23]+x[24]==1)
p.add_constraint(x[31]+x[32]+x[33]+x[34]==1)
p.add_constraint(x[41]+x[42]+x[43]+x[44]==1)

# Todos los departamentos tienen que recibir algún alumno
p.add_constraint(x[11]+x[21]+x[31]+x[41]==1)
p.add_constraint(x[12]+x[22]+x[32]+x[42]==1)
p.add_constraint(x[13]+x[23]+x[33]+x[43]==1)
```

```
p.add_constraint(x[14]+x[24]+x[34]+x[44]==1)
```

```
bigM = 10000000000
```

```
p.set_objective(
3*x[11]+       0*x[12]+ bigM*x[13]+    6*x[14]+
bigM*x[21]+      x[22]+    4*x[23]+    5*x[24]+
4*x[31]+    bigM*x[32]+    5*x[33]+    7*x[34]+
2*x[41]+       0*x[42]+    2*x[43]+ bigM*x[44])
```

```
p.solve()
```

```
11.0
```

```
p.get_values(x)
```

```
{11: 0.0,
 12: 1.0,
 13: 0.0,
 14: 0.0,
 21: 0.0,
 22: 0.0,
 23: 0.0,
 24: 1.0,
 31: 1.0,
 32: 0.0,
 33: 0.0,
 34: 0.0,
 41: 0.0,
 42: 0.0,
 43: 1.0,
 44: 0.0}
```

Por lo tanto, la solución al segundo apartado en este caso será:

- El estudiante 1 será destinado al departamento 2 (Maternidad)
- El estudiante 2 será destinado al departamento 4 (Cirugía)
- El estudiante 3 será destinado al departamento 1 (Urgencias)
- El estudiante 4 será destinado al departamento 3 (Medicina)