Curso 24/25
Grado en Ingeniería Informática
Heurística

Práctica 1 de Heurística y Optimización

~ · ~

Nombre, apellidos y NIA	
Alumn@ 1:	Ana Claver Miranda (100475965) Grupo 80 - Doble Grado en Ingeniería Informática y ADE
Alumn@ 2:	Sofía Ferreras Entero (100498789) Grupo 80 - Doble Grado en Ingeniería Informática y ADE

Índice

Introducción

En este informe vamos a describir los modelos matemáticos que hemos desarrollado para optimizar dos problemas de transporte aéreo que se nos han propuesto.

En primer lugar, se nos pide optimizar la maximización de los beneficios de la venta de distintos tipos de billetes, teniendo en cuenta unos precios fijos para las tarifas y unas limitaciones, tanto de capacidad de peso como del número de asientos disponibles. Para ello, hemos creado restricciones sobre la cantidad mínima y máxima de billetes vendidos, así como de la capacidad total de cada avión. Para llevar a cabo la resolución inicial de este modelo usamos *LibreOffice* y luego verificamos el resultado del modelo con *GLPK*.

El segundo problema que hemos tenido que modelizar nos propone enfrentarnos a la asignación de slots de aterrizaje con el objetivo de minimizar los costes asociados a los retrasos de los vuelos. En este caso, contamos con restricciones como; la necesidad de asignar un slot único por avión o el cumplimiento de los límites de tiempo que se nos proporcionan en el enunciado. Para resolver este problema hemos realizado una modelización que posteriormente hemos implementado en *GLPK*.

En el último apartado de este documento realizamos un análisis de los resultados obtenidos, un estudio de las posibles modificaciones que se podrán realizar y una comparación de las herramientas empleadas, además de una conclusión final.

Descripción de los modelos

A continuación, vamos a explicar las decisiones e interpretaciones que hemos adquirido del enunciado de la práctica, lo que nos ha llevado a la posterior modelización del problema, que se enuncia a continuación:

En primer lugar, la interpretación de la <u>función objetivo</u>:

Maximizar el beneficio total de la venta de billetes para cada avión. Puesto que sabemos que cada avión lleva un precio asignado a cada uno de los tipos de billetes, esta ecuación sería la siguiente: $max\ z\ = \sum_{j=1}^{T} P_j \cdot \sum_{i=1}^{A} x_{ij}\ \forall\ i \in AV;\ \forall\ j \in T$

En segundo lugar, la interpretación de las restricciones a tener en cuenta:

1) *Restricción de asientos máximos*: No se podrán vender más billetes que la cantidad de asientos disponibles en cada avión
2) *Restricción de capacidad del equipaje:* La suma del peso del equipaje permitido por los billetes no puede superar la capacidad máxima de carga de cada avión
3) *Restricción de billetes mínimos*: Para cada avión se deben ofertar como mínimo:
    3.1 20 billetes de *Leisure Plus*
    3.2 10 billetes de *Business Plus*
4) *Restricción de billetes de la tarifa Estándar:* Al menos el 60% de todos los billetes ofertados debe ser el número de billetes totales de la tarifa *Estándar*.

NOTA: En la modelización hemos indicado cada restricción con el número que le corresponde en este apartado.

1. Modelización a mano

Primero realizamos la modelización a mano en base a las restricciones que hemos sacado de las interpretaciones del enunciado de la práctica.

Datos:

AV_i: *Representa los aviones disponibles*

$\{AV_1 = Avión\ 1\ ,\ AV_2 = Avión\ 2\ ,\ AV_3 = Avión\ 3,\ AV_4 = Avión\ 4,\ AV_5 = Avión\ 5\}$

T: *Tarifas disponibles*

$\{T_1 = Estándar,\ T_2 = Leisure,\ T_3 = Business\}$

C_i: *Capacidad de los aviones disponibles* $\forall\ i \in AV$

$\{C_1 = 1700,\ C_2 = 2700,\ C_3 = 1300,\ C_4 = 1700,\ C_5 = 2000)$

P_j: *Precio de cada tarifa* $\forall j \in T$

$\{P_1 = 19,\ P_2 = 49,\ P_3 = 69)$

A_i: *Número de asientos disponibles en cada avión* $\forall\ i \in AV$

$\{A_1 = 90,\ A_2 = 120,\ A_3 = 200,\ A_4 = 150,\ A_5 = 190)$

E_j: *Equipaje permitido según la tarifa* $\forall\ j \in T$

$\{E_1 = 1,\ E_2 = 20,\ E_3 = 40\}$

Modelización:

$$max\, z = \sum_{j=1}^{T} P_j \cdot \sum_{i=1}^{A} x_{ij} \ \forall\, i \in AV;\ \forall\, j \in T$$

s.a.

$$\begin{cases} 1.\ \sum_{i=1}^{A} \sum_{j=1}^{T} x_{ij} \le A_i \ \ \forall \ i \in AV;\ \forall \ j \in T \\ 2.\ \sum_{i=1}^{A} \sum_{j=1}^{T} E_j \cdot x_{ij} \le C_i \ \ \forall \ i \in AV;\ \forall\, j \in T \\ 3.\ x_{ij} \begin{cases} \ge 20\, si\, j = 2 \ \forall \ j \in T \\ \ge 10\, si\, j = 3;\ \forall \ i \in AV;\ \forall \ j \in T \end{cases} \\ 4.\ \sum_{i=1}^{A} X_{i1} \ge 0,6 \cdot \sum_{i=0}^{A} \sum_{j=0}^{T} x_{ij} \ \ \forall \ i \in AV;\ \forall \ j \in T \end{cases}$$

2. Hoja de cálculo del modelo

Para resolver el problema hemos trabajado con la herramienta *LibreOffice*. En el documento .ods se podrán ver las siguientes columnas:

- *Aviones*: siendo AV_i; $\forall\, i \in AV$
- *Estándar*: Representa el número de billetes vendidos de la tarifa "*Estándar*", en nuestra modelización equivaldría a x_1
- *Leisure*: Representa el número de billetes vendidos de la tarifa "*Leisure*", en nuestra modelización equivaldría a x_2
- *Business*: Representa el número de billetes vendidos de la tarifa "*Business*", en nuestra modelización equivaldría a x_3
- *Total*: Realiza la suma de los números de billetes vendidos de cada tipo de tarifa $(x_{i1} + x_{i2} + x_{i3})\ \forall\, i \in AV$ de cada avión.
- *Equipaje*: Contiene la limitación de peso especificado en la modelización como P_j; $\forall\, j \in T$.

NOTA: Originalmente, estas 5 columnas que acabamos de citar están en blanco. Para realizar los posteriores cálculos sobre ellas.

- *Capacidad*: Esta columna representa las limitaciones de capacidad proporcionadas por la *Tabla 2* del enunciado de la práctica.
- *Asientos*: Esta columna contiene la limitación del número de asientos de cada avión proporcionado por la *Tabla 2* del enunciado de la práctica.

La función de beneficios que queremos maximizar es: $\sum_{j=1}^{T} P_j \cdot \sum_{i=1}^{A} x_{ij}\ \forall\, i \in AV;\ \forall\, j \in T$

Para ello, creamos una tabla que está originalmente vacía. La primera columna multiplica los valores de la tarifa *Estándar* por 19, la segunda columna multiplica los valores de la tarifa *Leisure* por 49 y la tercera columna multiplica por 69 los resultados de la tarifa *Business*. La última columna son los resultados de la función de maximización para cada avión. **26190 (celda objetivo $J$18)** que es el resultado de la suma de los billetes óptimos a vender multiplicados por el precio de sus respectivas tarifas, es decir, sustituidos en la ecuación Z para cada avión.

Para que nos diesen los resultados hemos usado la herramienta "Solver", seleccionando las celdas $H$2:$J$6 que son las modificadas, en la que hemos metido dos restricciones: Una que incluye todas las limitaciones que trabajan con ≤ **($M$13:$M$22 ≤ $O$13:$O$22)** y otra que incluye todas las limitaciones de tipo ≥ **($Q$13:$Q$23 ≥ $S$13:$S$23)**. Además de esta agrupación, nos hemos asegurado de trabajar con *"Solver lineal de LibreOffice" y* hemos modificado las *Opciones* que nos proporciona "*Solver*" para que considere todas la variables como <u>enteros</u> además de <u>no negativos</u>*.* Por último, hemos utilizado la fórmula de autosuma que ofrece esta herramienta con el fin de que, en caso de necesidad, modificar este documento resulte lo más sencillo posible.

3. GLPK (1ª parte)

Nuestro código en *GLPK* se adapta para determinar la cantidad de billetes que se ofertan con el fin de maximizar los ingresos de la compañía, ajustándose a las restricciones de cada avión, como el número de asientos y la capacidad de peso, además de cumplir con las restricciones mínimas del número de billetes vendidos por tarifa. A continuación expondremos las decisiones que hemos tomado y cómo hemos decido afrontar el problema:

Sets:

Los sets definen en nuestro código el conjunto de aviones y tarifas que usaremos a la hora de calcular el número de billetes óptimos para maximizar el beneficio.

1. **AVIONES**: Agrupa el conjunto de 5 aviones.
2. **TARIFAS**: Incluye las tres tarifas que se ofertan siendo *Estándar* (T1), *Leisure Plus* (T2) y *Business Plus* (T3). De esta forma, las variables y sus respectivos parámetros pueden referirse a cada tarifa de forma más intuitiva.

Parámetros:

Hemos decidido trabajar con los siguientes parámetros porque facilitan la implementación de las restricciones y el cálculo de la función objetivo que debemos maximizar, tomando en consideración los precios, las capacidades y las proporciones mínimas de venta.

1. **numero_asientos y capacidad**: Son nuestras restricciones en referencia al número de asientos y el peso del equipaje.
2. **min_billetes_2 y min_billetes_3:** Indican los billetes mínimos que se deben vender en las tarifas de *Leisure Plus* y *Business Plus* para cada avión, respectivamente.
3. **porcentaje**: Indica que al menos el 60% de los billetes vendidos deben ser de tarifa *Estándar*.
4. **precios y peso:** Estos parámetros los usamos como ayuda para calcular el beneficio y el peso total del equipaje de acuerdo a la tarifa que estemos estudiando.

Variables de decisión:

Ajustándose a las restricciones del problema, nuestra intención es obtener la combinación óptima de avión y tarifa que maximice los ingresos de la compañía.

billetes: Representa el número de billetes vendidos por avión y tarifa. (Como ya hemos indicado, es una variable entera)

Función objetivo:

Calcula el ingreso total de los billetes vendidos multiplicando el número de billetes de cada tarifa por su respectivo precio. De esta forma, obtenemos la mejor combinación de billetes para cada avión, maximizando así el beneficio.

Restricciones:

Contamos con las mismas 4 restricciones que en nuestro modelo. Estas las hemos introducido con s.t. (subject to)

1. **Restricción de asientos** *(restriccion_sitios_max)*: Limita el total de billetes vendidos en cada avión de acuerdo al número de asientos con los que cuenta.
2. **Restricción de peso** *(restriccion_capacidad_max):* Comprueba que el peso total del equipaje permitido para los billetes no se exceda de acuerdo a la capacidad máxima de cada avión.
3. **Restricciones de mínimos en las tarifas** *(restriccion_min_billetes_2; restriccion_min_billetes_3):* Se asegura de que se vendan por lo menos 20 billetes de la tarifa *Leisure Plus* (T2) y 10 billetes de la tarifa *Business Plus* (T3) por avión.
4. **Restricción tarifa Estándar** *(restriccion_min_oferta_60)*: Fuerza a que al menos el 60% de todos los billetes vendidos sean de tarifa *Estándar* (T1).

Como resultado hemos obtenido el mismo beneficio que en la hoja de cálculo, **26190**, pero con diferentes resultados de las variables que estudiamos (los números de billetes de cada tarifa) lo que nos hace concluir que existe más de una solución en la región factible que conlleva el mismo resultado óptimo.

4. Modelización a mano (2ª parte)

Para la segunda parte de la práctica, tratamos de resolver el problema de la distribución de las pistas de aterrizaje de acuerdo a los slots que se nos proporcionan en la *Figura 1*. A continuación desarrollamos las interpretaciones que hemos obtenido del enunciado:

En primer lugar, la interpretación de la <u>función objetivo</u>:

Minimizar el coste por retraso en la asignación de los aviones a los slots. Por lo tanto, nuestra función objetivo consiste en la multiplicación del retraso de los aviones, por el coste correspondiente el avión que ha sufrido el retraso:

$$min\ Z = \sum_{i=1}^{A} \sum_{j=1}^{S} \sum_{k=1}^{P} (HS_j - HA_i) \cdot C_i \cdot x_{ijk} \quad \forall i \in A;\ \ \forall j \in S;\ \forall\ k \in P$$

En segundo lugar la interpretación de las <u>restricciones</u> a tener en cuenta:

1) Cada avión debe tener asignado un slot de tiempo para efectuar el aterrizaje

1.1) Sólo puede haber un avión en cada slot de tiempo, esta restricción la hemos modelado de la siguiente manera $\sum_{i=0}^{A} x_{ijk} \leq 1 \, \forall \, i \in AV$. Posteriormente, nos hemos dado cuenta de que esta restricción no aporta nada matemáticamente, puesto que al ser x_{ijk} una variable binaria el valor de esta será siempre , por lo que no la incluidos en la modelización ya que es redundante.

2) El slot que se asigna debe estar disponible.
3) El inicio del slot de aterrizaje debe ser igual o posterior a la hora de llegada del avión así como igual o menos a la hora límite de aterrizaje del avión.
4) No se pueden asignar dos slots consecutivos en la misma pista.

NOTA: En la modelización hemos indicado cada restricción con el número que le corresponde en este apartado.

Datos:

A: $Representa\ los\ aviones\ disponibles\ \{AV_1, AV_2, AV_3, AV_4, AV_5\}$

P: $Pistas\ de\ aterrizaje\ \{P1, P2, P3, P4,\}$

S: $Slots\ \{s1, s2, s3, s4, s5, s6\}$

C_i: $Coste\ por\ cada\ minuto\ que\ se\ retrase\ el\ vuelo\ \ \forall i \in A$

$\{C_1 = 100,\ C_2 = 200,\ C_3 = 150,\ C_4 = 250,\ C_5 = 200\}$

HL_i: $Hora\ límite\ de\ aterrizaje\ \ \forall i \in A$

$\{HL_1 = 10{:}15,\ HL_2 = 9{:}30,\ HL_3 = 10{:}00,\ HL_4 = 10{:}15,\ HL_5 = 10{:}30\}$

HI_j: $Hora\ de\ inicio\ del\ slot\ \ \forall j \in S$

$\{HI_1 = 9{:}00,\ HI_2 = 9{:}15,\ HI_3 = 9{:}30, HI_4 = 9{:}45,\ HI_5 = 10{:}00,\ HI_6 = 10{:}15\}$

HA_i: $Hora\ de\ aterrizaje\ del\ avión\ \ \forall i \in A$

$\{HA_1 = 9{:}10,\ H_2 = 8{:}55,\ H_3 = 9{:}40,\ H_4 = 9{:}55,\ H_5 = 10{:}10\}$

$$x_{ijk} \begin{cases} 1:\ si\ el\ avión\ i\ aterriza\ en\ el\ slot\ j\ en\ la\ pista\ k \quad \forall i \in A;\ \forall j \in S;\ \forall\ k \in P \\ 0:\ si\ el\ avión\ \ i\ no\ aterriza\ en\ el\ slot\ j\ en\ la\ pista\ k \quad \forall i \in A;\ \forall j \in S;\ \ \forall\ k \in P \end{cases}$$

Modelización:

$$min\, Z = \sum_{i=1}^{A} \sum_{j=1}^{S} \sum_{k=1}^{P} (HS_j - HA_i)\cdot C_i \cdot x_{ijk} \quad \forall i \in A; \ \ \forall j \in S; \ \forall\, k \in P$$

$$\text{s.a.} \begin{cases} 1.\ \sum_{j=1}^{S} \sum_{k=1}^{P} x_{ijk} = 1 \quad \forall i \in A; \ \ \forall j \in S; \ \forall\, k \in P \\ 2.\ \quad x_{ijk} \le \begin{cases} 0\ si\ j \in & \forall i \in A; \ \ \forall j \in S; \ \forall\, k \in P \\ 1\ si\ j \notin\ ST & \forall i \in A; \ \ \forall j \in S; \ \forall\, k \in P \end{cases} \\ 3.\ (x_{ijk} \cdot (HA_i \le HI_j \le HL_i)) \le 1\, \forall i \in A; \ \ \forall j \in S \\ 4.\ (x_{ijk} + \sum_{m=1}^{A} x_{mj+1,k} + \sum_{m=1}^{A} x_{mj-1,k}) \le 1 \forall i \in A; \ \ \forall j \in S; \ \forall\, k \in P \end{cases}$$

5. GLPK (2ª parte)

Para el segundo problema hemos ideado un código en *GLPK* que optimiza la asignación de slots de tiempo y pistas de aterrizaje con el fin de minimizar el coste que resulta de que haya retrasos en los aterrizajes, siempre respetando las restricciones de disponibilidad de pistas y slots. Nos gustaría hacer hincapié en que nuestros modelos de GLPK son muy generales, lo que nos ha facilitado hacer las pruebas a la hora de cambiar datos ya que solo haría falta recurrir al archivo .dat. Para ello hemos declarado los siguientes elementos y hemos tomado las siguientes decisiones:

Sets:

Hemos decidido usar la separación en conjuntos (AV, ST, PT) para simplificar el modelo ya que nuestro objetivo es reducir la complejidad y que el código sea de fácil interpretación.

1. **AV**: Representa los aviones a los que hay que asignar un slot y una pista de aterrizaje.
2. **ST**: Representa los slots de tiempo que están disponibles.
3. **PT**: Representa las pistas de aterrizaje.

4. **validos_para_pistas**: Asigna a cada pista los slots que específicamente están disponibles para ella. Para ello tenemos los siguientes subconjuntos que indican la disponibilidad de los slots para cada pista: aterrizar_PT1, aterrizar_PT2, aterrizar_PT3, aterrizar_PT4.

Parámetros:

Decidimos usar los parámetros que citamos a continuación para aportar flexibilidad en el ajuste de tiempos y costes, sin tener que modificar el modelo del que partimos. También comentaremos el uso del número "M".

1. **Coste{AV}**: Representa el coste por minuto de retraso para cada avión.
2. **HL{AV}**: Es la hora límite de aterrizaje para cada avión. Pasada esta hora, el avión no puede aterrizar.
3. **HI{ST}**: Define la hora de inicio de cada slot de tiempo.
4. **HA{AV}**: Incluye las horas de aterrizaje de cada avión.
5. **sig_slot y ant_slot:** Estos parámetros los hemos creado específicamente para la restricción de que no se pueden asignar dos slots consecutivos. De esta forma, relacionamos los slots entre sí, identificando el slot anterior y posterior.
6. **M = 10000:** Lo hemos creado al estar trabajando con variables binarias para penalizar soluciones no deseadas en la función objetivo.

Variables de decisión:

Solo empleamos una variable de decisión que es binaria, lo que nos permite modelar de forma sencilla si un avión aterriza en un slot específico o no.

1. **aterriza_si_o_no{AV, ST, PT} binary:** En caso de que un avión aterrice en un slot y pista específicos vale 1, en caso contrario valdrá 0.

Función objetivo:

En esta función (*perdidas*) tomamos en consideración el coste por minuto de retraso para cada avión, buscando minimizar los costes. Para ello multiplicamos el coste por minuto (Coste[i]) por la diferencia entre el inicio del slot (HI[j]) y la hora de aterrizaje de cada avión (HA[i]).

Restricciones:

tiene_slot: Con esta restricción nos aseguramos de que cada avión se asigne a un único slot y pista a la hora de aterrizar .

consecutivos_arriba y consecutivos_abajo: La primera restricción nos asegura que el avión no aterrice antes de su hora de llegada (*HA[i]*) y la segunda asegura que no lo haga después de su hora límite (*HL[i]*)
no_slots_consecutivos: Si un avión aterriza en un slot, ningún otro avión podrá aterrizar ni en el periodo del slot anterior ni el periodo posterior.
slot_disponible_para_aterrizar: Verifica que el slot k esta dentro de los slots válidos para la pista que le corresponde (validos_para_pistas[k])

6. GLPK (Parte conjunta)

Para el *GLPK* conjunto hemos juntado el código de la parte 1 con el código de la parte 2. Siendo la función objetivo la ecuación de ***Ingresos = Beneficios*** (función objetivo de la parte 1) **- *Costes*** (función objetivo de la parte 2). Todas las restricciones se mantienen idénticas al igual que los parámetros y los sets. Además, todas las explicaciones de cada uno de los elementos del código sigue siendo idéntica. El resultado de la función objetivo es Ingresos = Beneficios(26190) - Coste(4500) = **21690**.

Nos gustaría hacer hincapié en que nuestros modelos de *GLPK* son muy generales lo que nos ha facilitado hacer las pruebas a la hora de cambiar datos ya que solo haría falta recurrir al archivo .dat.

Análisis de los resultados

Tras realizar la resolución del problema, tanto en *GLPK* como con *LibreOffice*, hemos obtenido el mismo resultado en ambos casos, siendo este un beneficio de 26190 euros. Esto nos demuestra que nuestro problema de programación lineal ha llegado en ambos casos al resultado óptimo que deseábamos, pero con diferentes soluciones para cada avión, por lo que concluimos que hay **soluciones óptimas alternativas**. A continuación evaluaremos su complejidad.

Para la segunda parte, que llamamos parte conjunta. Hemos realizado en primer lugar el código únicamente de la parte 2 cuya función objetivo evalúa los costes. A continuación, hemos juntado las funciones objetivo y las restricciones haciendo una nueva función objetivo. En el siguiente apartado calculamos la complejidad del primer problema y las pruebas y modificaciones de esta parte, y luego la complejidad de la parte conjunta.

Complejidad parte 1:

A continuación, para evaluar la complejidad del primer problema vamos a tomar en consideración el número de restricciones de variables que hemos necesitado:

Variables: Para cada avión y tarifa, tenemos una variable x_{ij} que representa el número de billetes vendidos para el avión i con la tarifa j. Ya que contamos con 5 aviones y 3 tarifas esto implica que hemos trabajado con 15 variables.

Restricciones: Como hemos indicado con anterioridad, contamos con 4 restricciones:

1. No se puede exceder el número máximo de asientos disponibles por cada avión, por lo que tenemos 1 restricción por cada elemento del conjunto AV_i (5 restricciones)
2. El equipaje no puede exceder el valor máximo de capacidad de peso del avión, por lo que, de nuevo, tenemos 1 restricción por cada elemento del conjunto AV_i (5 restricciones)
3. Se deben vender por lo menos 20 billetes de la tarifa *Leisure* y 10 billetes de la tarifa *Business*. Debido a que ambas tarifas aparecen en cada avión contamos con 2 restricciones por avión (10 restricciones)
4. El número de billetes vendidos de la tarifa *Estándar* debe ser por lo menos el 60% del sumatorio de los billetes vendidos: Esto resulta en una restricción global que no depende del número de aviones que estudiamos. (1 restricción global)

En total, contamos con 21 restricciones.

Pruebas y posibles modificaciones de restricciones:

1. ***Cambiar los parámetros de capacidad o los precios de las tarifas:*** Estas posibilidades no afectarán a la complejidad del problema pero sí varía el resultado de los billetes y consecuentemente el beneficio total.
2. ***Añadir o eliminar aviones:*** En el caso de añadir más aviones se incrementa el número de variables y restricciones linealmente, por lo que aumenta la complejidad del problema al haber más limitaciones que cumplir. Y aumentará la función objetivo en consecuencia.

3. ***Cambiar las limitaciones de las tarifas:*** Tanto como si se aumenta el mínimo obligatorio de las tarifas o el porcentaje de billetes *Estándar* requerido se restringiría más la solución, lo que potencialmente reduciría el beneficio máximo.

Complejidad parte conjunta:

A continuación evaluaremos la parte conjunta del código:

Variables: Para cada avión, slot y pista, tenemos una variable X_{ijk}que representa 5 aviones, 6 slots y 4 pistas. Esto implica que hemos trabajado con 120 variables.

Restricciones: Como hemos indicado con anterioridad, contamos con 4 restricciones de la parte 1 cuyo total son (21 restricciones) .

5. Sólo puede haber un avión por cada pista por lo que ⇒ 5 aviones · 4 pistas. (20 restricciones).
6. El slot que se asigna debe estar disponible para cada slot de tiempo por lo que⇒ 5 aviones · 6 slots de tiempo. (30 restricciones).
7. La tercera restricción de la modelización la hemos divido en dos parte como explicamos en el apartado de *GLPK* por lo que ⇒2 restricciones(5 aviones ·6 Slots). (60 restricciones).
8. No se pueden asignar dos slots consecutivos en la misma pista ⇒ 5 aviones · 6 slots·4 pistas. (120 restricciones).

120 restricciones de la parte 1 + 230 restricciones = 350 restricciones en total.

Pruebas y posibles modificaciones:

1. Para evaluar los retrasos en el modelo hemos modificado el *.dat*. Para ponerlo a prueba, hemos puesto el valor de HI_1 cómo 540 y el valor de HA_1 cómo 520 y el resultado nos ha aumentado a 6000. Al poner 510 en vez de 520 en HA_1 aumenta a 7000. Por lo que concluimos que el aumento del tiempo del retraso es directamente proporcional al resultado.
2. En caso de haber “overbooking”, es decir, vendiendo más billetes que la capacidad máxima de aviones. (Nuestra variable 1 de la modelización 1 sería eliminada en este caso.) Los ingresos de la empresa aumentaron por lo que la función objetivo aumenta. Siendo ahora de 106.450€.
3. En caso de haber vendido más billetes que superen la capacidad máxima de los aviones (Eliminando la 2ª variable de la modelización 1). También

aumentará el beneficio de la empresa, pero lo más seguro es que haya pérdidas de equipaje.

Limitaciones de las herramientas:

En primer lugar, *LibreOffice* cuenta con una interfaz visual muy intuitiva que facilita la organización de los datos, lo que consideramos clave para un problema como este, y permite crear modelos que no cuenten con muchas variables sin necesidad de conocimientos avanzados en el uso de esta aplicación. Sin embargo, este programa podría resultar problemático en el caso de enfrentarse a problemas más grandes que cuenten con mayor complejidad.

Por otro lado, *GLPK* es una herramienta que parece específicamente diseñada para problemas de programación lineal, por lo que es preferible usarla para resolver problemas de optimización con múltiples restricciones y un gran número de variables. Pero, hay que considerar que a diferencia de *LibreOffice*, que resulta intuitivo y fácil de usar, *GLPK* requiere ciertos conocimientos de programación y modelización de problemas matemáticos.

Conclusiones:

Para concluir nos gustaría hacer unos comentarios sobre la práctica. Nos hemos encontrado muy satisfechas con esta práctica, tanto en la realización que hemos hecho, como con el enunciado propuesto. Consideramos que es muy buena idea que las prácticas propuestas se basen en problemas que se pueden dar en la vida real, haciendo que adaptemos estas situaciones a lo aprendido en clase y los conocimientos adquiridos. Entre ellos, aprender el lenguaje de *GLPK* y el uso de los .dat y .mod, el uso de herramientas de *LibreOffice* y la modelización de problemas heurísticos.