Práctica: *Programación Lineal.*
Heurística y Optimización
Grado de Ingeniería en Informática. Curso 2017-2018.

Planning and Learning Group
Departamento de Informática

1. Objetivo

El objetivo de esta práctica es que el alumno aprenda a modelar problemas de programación lineal y a resolverlos con dos tipos de herramientas: hojas de cálculo y algoritmos de resolución usando técnicas de modelización.

2. Enunciado del problema

Una importante compañía aérea de transporte de mercancías y pasajeros con sede en Madrid se ha puesto en contacto con alumnos del curso de Heurística y Optimización. Tras varias entrevistas os han seleccionado a tu compañero y a ti para dar solución a varios de sus problemas mediante el uso de técnicas de programación lineal.

2.1. Parte 1: Modelo básico en Calc

La compañía para la que ahora trabajamos nos explica que existe un grave problema a la hora de almacenar el equipaje de mano de los pasajeros. Nos cuenta que en muchas ocasiones los pasajeros colocan mal su equipaje en los compartimentos destinados para ello, ocupando mucho más espacio del que les corresponde. Este uso ineficiente del espacio disponible hace que las maletas de algunos pasajeros no quepan, obligando a la compañía a trasladarlas a la bodega del avión con la consecuente pérdida de tiempo y dinero. Por este motivo, se ha decidido automatizar la colocación del equipaje de mano en los compartimentos, de forma que se evite o al menos minimice el traslado de maletas a la bodega.

Para ello, nos informa que ha agrupado el equipaje de mano que se va a cargar dentro de un determinado vuelo en tres categorías, M1, M2 y M3, como se muestra en la Tabla 1.

Categoría	Número	Peso (kg)	Dimensiones (cm)	Coste (€)
M1	22	7	30×20×10	10
M2	18	8	40×20×10	20
M3	11	10	50×30×20	30

Tabla 1: Categoría del equipaje de mano.

La segunda columna de la Tabla 1 muestra el número de maletas dentro de cada categoría que hay que embarcar en el avión. La tercera y cuarta columna muestran el peso y las dimensiones ($alto \times largo \times ancho$) de cada maleta dentro de la categoría correspondiente. La última columna muestra el coste/maleta por tener que trasladarla a la bodega del avión.

Además, nos informa que el avión dispone de seis compartimentos para situar todo este equipaje y se distribuyen como se muestra en la Figura 1. Cada uno de estos compartimentos puede almacenar como máximo el peso y volumen que se muestran en la Tabla 2. En cada compartimento se pueden almacenar maletas de cualquier categoría.

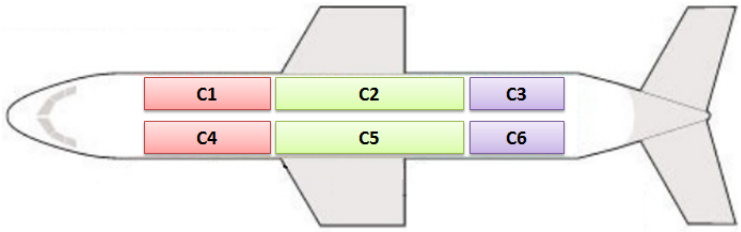

Figura 1: Distribución de los compartimentos dentro del avión.

Compartimento	Peso (kg)	Volumen (m^3)	Compartimento	Peso (kg)	Volumen (m^3)
C1	50	0.1	C4	50	0.1
C2	60	0.15	C5	60	0.15
C3	40	0.07	C6	40	0.07

Tabla 2: Peso y volumen total que pueden almacenar los compartimentos.

Por último, la compañía nos informa que el avión debe tener el centro de gravedad adelantado puesto que con ello se consigue una mayor estabilidad y una mejor recuperación si el avión entra en pérdida. Para conseguirlo, el peso del equipaje de los compartimentos C1 y C4 debe ser superior al menos en un 10 % al peso del equipaje de los compartimentos C3 y C6.

El objetivo es determinar cuál es el número de maletas de cada categoría que se almacenan en cada uno de los compartimentos del avión, de forma que la compañía minimice el gasto por trasladar las maletas que no cogen a la bodega.

Se pide:

1. Modelar el problema como un problema de programación lineal entera.
2. Implementar el modelo en una hoja de cálculo (LibreOffice).

2.2. Parte 2: Modelo avanzado en GLPK

La compañía aérea está satisfecha con los resultados que hemos obtenido en la primera parte, y ahora nos plantea otro problema que desea resolver: la asignación de tripulantes (pilotos y auxiliares) a los vuelos que tiene previstos en el día. La Tabla 3 muestra información sobre el origen y destino y los horarios de salida y llegada de cada uno de estos vuelos.

Vuelo	Trayecto	Salida	Llegada
V1	Madrid-Valencia	8:30	9:45
V2	Valencia-Madrid	10:30	12:00
V3	Madrid-Valencia	13:00	14:45
V4	Valencia-Madrid	15:00	16:05
V5	Madrid-Valencia	17:45	19:00
V6	Valencia-Madrid	19:50	21:00

Tabla 3: Información sobre los vuelos.

El primer vuelo, V1, se corresponde con el vuelo de la primera parte y por tanto en esta parte también habrá que considerar el gasto que se produce por trasladar las maletas sobrantes de los pasajeros a la bodega del avión. El resto son vuelos de carga que no llevan pasajeros y por tanto no requieren trasladar ninguna maleta.

Para realizar los vuelos anteriores, la compañía dispone de tres pilotos y tres auxiliares. Sobre cada uno de estos pilotos y auxiliares la compañía nos proporciona la información sobre el dinero que quieren cobrar por cada hora de vuelo en cada uno de los vuelos anteriores. Esta información se muestra en las Tablas 4 y 5.

Sobre la asignación de los tripulantes a los vuelos, la compañía nos impone las siguientes restricciones. Lógicamente cada vuelo debe tener asignado al menos un piloto, y al menos un auxiliar. Además, la compañía

Piloto	V1	V2	V3	V4	V5	V6
P1 (€/h)	20	17	25	34	31	22
P2 (€/h)	28	15	23	35	37	21
P3 (€/h)	27	16	24	31	35	29

Tabla 4: Sueldo por hora de vuelo que los pilotos desean cobrar en cada vuelo.

Auxiliar	V1	V2	V3	V4	V5	V6
A1 (€/h)	17	16	14	14	11	12
A2 (€/h)	16	14	12	15	17	11
A3 (€/h)	15	15	14	11	15	19

Tabla 5: Sueldo por hora de vuelo que los auxiliares desean cobrar en cada vuelo.

acaba de firmar un convenio con los trabajadores en el cual se garantiza que las horas totales de vuelo de los auxiliares será superior al de los pilotos. En este convenio también se recoge que cada piloto debe acordar con la compañía un tiempo de descanso mínimo entre dos vuelos que debe realizar de forma consecutiva, i.e., un tiempo mínimo de descanso entre la llegada de un vuelo que pilota y la salida del siguiente que debe pilotar. En este caso, el tiempo mínimo acordado por los pilotos considerados son de 60 minutos para el piloto P1, 25 minutos para el piloto P2 y 30 minutos para el piloto P3. Por último, hay que tener en cuenta que un tripulante (piloto o auxiliar) no puede ser asignado a un vuelo si no se encuentra en el aeropuerto de origen de ese vuelo. Sobre este hecho la compañía nos informa que inicialmente todos los pilotos y auxiliares están situados en el aeropuerto de Madrid y, por lo tanto, no pueden ser asignados a un vuelo con origen Valencia si previamente no han volado a esa ciudad.

El objetivo en esta parte es minimizar el gasto generado por la compañía por el traslado de maletas a la bodega en el primer vuelo V1, y por la asignación de pilotos y auxiliares a cada uno de los vuelos. Para ello, hay que tener en cuenta las restricciones impuesta por la compañía.

Se pide:

1. Modelar el problema como un problema de programación lineal entera.

2. Implementar el modelo en un lenguaje de modelado más sofisticado (MathProg).

2.3. Parte 3: Análisis de Resultados

En este apartado se deben analizar todos los resultados obtenidos, describir la solución obtenida (comprobando que cumple con las restricciones de este enunciado) y analizar qué restricciones están limitando la solución del problema.

Análisis de la complejidad del problema: ¿cuántas variables y restricciones has definido?, modifica el problema, variando los parámetros o añadiendo/eliminando maletas, compartimentos, vuelos, tripulantes y explica cómo esto afecta a la dificultad de resolución del problema resultante.

Además, se debe contestar la siguiente cuestión: ¿cuál es el número de pilotos y auxiliares que se asigna a cada vuelo? ¿Por qué consideras que determinados vuelos tienen más de un piloto y un auxiliar aunque ello implica un mayor coste para la compañía?

Asimismo, discute las ventajas y desventajas de las dos herramientas utilizadas en la asignatura: LibreOffice y GLPK.

2.4. Parte extra: Problema de Programación Dinámica

Dado el éxito que ha supuesto el modelo propuesto para la empresa, al cabo de los días se pone en contacto con nosotros nuevamente para solucionar otro tipo de problema. Nos informa que está pensando abrir una nueva ruta comercial en las que visitará las ciudades de Vigo, San Sebastián, Sevilla, Córdoba y Barcelona. Para realizar esta ruta quieren emplear un único avión, puesto que considera que eso será suficiente para mover el volumen de carga que inicialmente se plantea entre estas ciudades. La ruta partirá de Madrid, visitará todas las

ciudades previstas una única vez, y regresará nuevamente a Madrid. La compañía nos pide realizar un programa mediante programación dinámica que decida cuál debería ser el itinerario a seguir para minimizar la longitud de la ruta. El programa recibirá por parámetro un fichero que contendrá la distancia entre todas las ciudades consideradas. A continuación se muestra un ejemplo de ese fichero:

```
    M   V   Ss  Sv  C   B
M   0   602 450 434 394 624
V   602 0   755 741 796 1152
Ss  450 755 0   916 841 573
Sv  434 741 916 0   140 998
C   394 796 841 140 0   865
B   624 998 573 998 865 0
```

Para realizar pruebas, se recomienda generar ficheros con diferente número de ciudades. Se debe tener en cuenta que la distancia de una ciudad a sí misma debe ser cero, por lo que la diagonal principal de la matriz debe ser nula. Deberá entregarse el programa en el lenguaje que se desee, junto a un script en bash que lo ejecute: `ruta.sh`. Dicho script recibirá como argumento el fichero de distancias anterior e imprimirá por pantalla 2 líneas: la primera indicando la distancia total de la ruta y la segunda el itinerario generado. A continuación se muestra un ejemplo de fichero de salida:

```
Distancia total:    3338
Ruta:   M,Sv,C,V,Ss,B,M
```

En la memoria deberán documentarse las ecuaciones de programación dinámica empleadas, así como posibles optimizaciones adicionales que ayuden a que el programa sea más eficiente. Además, deben contestarse las siguientes preguntas:

- Si se deben visitar 100 ciudades diferentes, ¿cuál es el número de combinaciones que haría un algoritmo de fuerza bruta?. ¿cuál es el número de combinaciones que considera tu solución?
- ¿Sería más eficiente la ejecución del algoritmo si los ciudades estuvieran ordenadas en función de algún criterio?

3. Directrices para la Memoria

La memoria debe entregarse en formato .pdf y tener un máximo de 15 hojas en total, incluyendo la portada, contraportada e índice. Al menos, ha de contener:

1. Breve introducción explicando los contenidos del documento.
2. Descripción de los modelos, argumentando las decisiones tomadas.
3. Análisis de los resultados.
4. Conclusiones acerca de la práctica.

La memoria **no debe incluir código fuente** en ningún caso.

4. Evaluación

La evaluación de la práctica se realizará sobre 10 puntos. Para que la práctica sea evaluada deberá realizarse al menos el apartado 1 y la memoria. La distribución de puntos es la siguiente:

1. Parte 1 (3 puntos)

- Modelización del problema (1 punto)
- Implementación del modelo (2 puntos)

2. Parte 2 (5 puntos)
   - Modelización del problema (3 puntos)
   - Implementación del modelo (2 puntos)
3. Parte 3 (2 puntos)
4. Parte Extra: Programación Dinámica (1 punto)

En la evaluación de la modelización del problema, un modelo correcto supondrá la mitad de los puntos. Para obtenerse el resto de puntos, la modelización del problema deberá:

- Ser formalizada correctamente en la memoria.
- Ser, preferiblemente, sencilla y concisa.
- Estar bien explicada (ha de quedar clara cuál es la utilidad de cada variable/restricción).
- Justificarse en la memoria todas las decisiones de diseño tomadas.

En la evaluación de la implementación del modelo, un modelo correcto supondrá la mitad de los puntos. Para obtenerse el resto de puntos, la implementación del problema deberá:

- Hacer uso (en la implementación) de las capacidades que ofrecen las herramientas para que hacer/actualizar el modelo sea lo más sencillo posible (por ejemplo, utilizar `sumaproducto` si es posible en el caso de la hoja de cálculo o el uso de `sets` en MathProg).
- Mantener el código (hoja de cálculo o ficheros de MathProg) correctamente organizado y comentado. Los nombres deben ser descriptivos. Deberán añadirse comentarios en los casos en que sea necesario para comprenderlo.

Al puntuar el análisis de resultados, se valorará positivamente el hecho de incluir en la memoria conclusiones personales acerca de la dificultad de la práctica y de lo aprendido durante su elaboración.

Importante: los modelos implementados en la hoja de cálculo y GLPK deben ser correctos. Esto es, han de funcionar y obtener soluciones óptimas al problema que se solicita. En ningún caso se obtendrá una calificación superior a 1 punto por un modelo que no es correcto. Por tanto, si la parte 1 no está correctamente acabada, la nota máxima será de 1 punto y, si la parte 2 no está correctamente acabada, como mucho se obtendrá una calificación de 4 puntos.

5. Entrega

Se tiene de plazo para entregar la práctica hasta el 29 de Octubre a las 23:55. Sólo un miembro de cada pareja de estudiantes debe subir a la sección de prácticas de Aula Global un único fichero `.zip`. Es importante cumplir las siguientes instrucciones para la entrega.

El fichero debe nombrarse `p1-NIA1-NIA2.zip`, donde `NIA1` y `NIA2` son los últimos 6 dígitos del NIA (rellenando con 0s por la izquierda si fuera preciso) de cada miembro de la pareja. Ej: `p1-054000-671342.zip`.

La descompresión de este fichero debe producir al menos dos directorios llamados exactamente "`parte-1`" y "`parte-2`". Además, si se entregara la parte extra (de Programación Dinámica), entonces debe incluirse un tercer directorio "`parte-3`" que la contenga.

La memoria en formato .pdf debe incluirse en la raíz de estos directorios y debe llamarse `NIA1-NIA2.pdf`, mientras que las soluciones (ya sea código fuente, o una hoja de cálculo) de cada parte deben incluirse en el subdirectorio que corresponda.

Importante: no seguir las normas de entrega puede suponer una pérdida de hasta 1 punto en la calificación final de la práctica.