Universidad de La Laguna

SECCIÓN DE MATEMÁTICAS | FACULTAD DE CIENCIAS

Zuleica Reina Segura

Optimización Combinatoria en la selección óptima de vuelos en la encuesta de turismo del ISTAC

Combinatorial optimization in the optimal selection of flights in the ISTAC tourism survey

Trabajo Fin de Grado
Grado en Matemáticas
La Laguna, Septiembre de 2018

Dirigido por

Juan José Salazar González

Juan José Salazar González
Departamento de Matemáticas, Estadística e Investigación Operativa
Universidad de La Laguna
38271 La Laguna, Tenerife

Agradecimientos

Quiero agradecer a mi tutor Juan José Salazar por su gran ayuda y apoyo durante la investigación ya que me ha aportado todas las herramientas necesarias para terminar mi trabajo de fin de grado satisfactoriamente.

También me gustaría agradecer la colaboración del ISTAC, en especial de Yenis González, que me ha aportado toda la información posible y me ha resuelto algunas dudas surgidas durante el desarrollo del trabajo.

Por último, darle las gracias a mi familia y amigos, en especial mi madre y mi compañera María Pérez, que han estado ahí para apoyarme incondicionalmente.

Resumen · Abstract

Resumen

La encuesta FRONTUR-Canarias realizada por el ISTAC pretende estudiar la demanda turística en Canarias. Para ello, se necesita un modelo matemático que resuelva el problema de maximizar el número de encuestas teniendo en cuenta que se debe incluir encuestas de todos los destinos posibles para obtener la mayor información posible. Además, tenemos que considerar varias limitaciones a la hora de resolver el problema. Para desarrollar el algoritmo que resuelve el problema implementamos un código en C y usamos el CPLEX para obtener las soluciones.

Palabras clave: *Encuesta FRONTUR – ISTAC – Modelo matemático – Algoritmo.*

Abstract

The FRONTUR-Canarias survey conducted by the ISTAC aims to study tourism demand in the Canary Islands. For this, a mathematical model is needed that solves the problem of maximizing the number of surveys taking into account that surveys of all possible destinations should be included to obtain as much information as possible. In addition, we have to consider several limitations when it comes to solving the problem. To develop the algorithm that solves the problem, we implement a code in C and use the CPLEX to obtain the solutions.

Keywords: *FRONTUR Survey – ISTAC – Mathematical model – Algorithm.*

Contenido

Introducción

El ISTAC realiza las estadísticas sobre la demanda turística que se recogen en relación con el número de viajes turísticos y el número de pernoctaciones en esos viajes. En este contexto se sitúa la Encuesta de Movimientos Turísticos en Fronteras en Canarias (FRONTUR). La muestra de FRONTUR-Canarias se compone de encuestas a pasajeros en vuelos nacionales e internacionales con origen un aeropuerto canario.

El objetivo de este trabajo de fin de grado es maximizar el número de encuestas que tiene que realizar el ISTAC considerando las siguientes restricciones:

* La jornada laboral de los encuestadores es de 8 horas como máximo.
* No se pueden encuestar los vuelos seleccionados por el INE.
* El número de encuestas está restringido a la tabla de tamaño muestral mínimo.
* En cada aeropuerto no puede haber más de dos encuestadores en el mismo día.
* Las encuestas se realizan durante 7 días consecutivos al mes.
* La ocupación de los aviones es del 80 % del total.
* Se tiene una tasa de encuesta del 50 % de los pasajeros de cada vuelo seleccionado.
* Cada encuesta se hace en un tiempo limitado de 30 segundos y debe finalizar cinco minutos antes de la salida del vuelo.
* Los aeropuertos canarios donde se realizan las encuestas son Tenerife Sur (TFS), Tenerife Norte(TFN), Gran canaria (LPA), Lanzarote (ACE), Fuerteventura (FUE), La Palma (SPC).

Para ello desarrollaremos un algoritmo matemático que resuelva este problema de manera óptima.

La selección de vuelos a encuestar la realiza una empresa contratada por el ISTAC, quien posteriormente planifica también el horario de los encuestadores. Nuestra labor es usar la programación matemática y demostrar que puede ser bastante útil para resolver este tipo de problemas.

En el primer capítulo explicaremos en qué consiste la encuesta FRONTUR, cuáles son sus principales objetivos y el método de selección de vuelos. Posteriormente, describimos el problema con sus limitaciones y proponemos el modelo matemático que optimiza el problema.

En el segundo capítulo comentamos varios problemas de programación lineal que poseen algoritmos y están relacionados con el que tenemos en particular.

El tercer y último capítulo muestra el desarrollo del trabajo de fin de grado en el cual implementamos un código en C y usando el modelo matemático explicado anteriormente, resolvemos el problema mostrando algunas soluciones de una semana en concreto y de los aeropuertos de Tenerife Norte, Lanzarote y Tenerife Sur. Por último, terminamos con unas conclusiones finales.

1

Descripción del problema de optimización vinculado al diseño de la encuesta

En primer lugar vamos a describir en qué consiste la encuesta FRONTUR y cuáles son sus objetivos. A continuación, explicaremos el método que se utiliza para seleccionar los vuelos que se quieren encuestar. Además, podremos ver una tabla con los tamaños mínimos muestrales que se necesitan para cada vuelo según el número de pasajeros del avión. Por último, se explica el problema que nos propone el ISTAC y podremos observar el modelo matemático que utilizamos para resolver dicho problema con sus restricciones.

1.1. Descripción de la encuesta

La Encuesta de Movimientos Turísticos en Fronteras en Canarias (FRONTUR-Canarias) se basa en las estadísticas sobre la demanda turística que se recogen en relación con el número de viajes turísticos y el número de pernoctaciones en esos viajes.

En el año 2015 el ISTAC suscribió un nuevo convenio de colaboración con el INE para la ampliación de la muestra de FRONTUR sobre turismo internacional en la Comunidad Autónoma de Canarias, con la finalidad de disponer de un tamaño muestral suficiente para desagregar la estadística por islas y por países de residencia. En ese sentido la muestra total del ámbito de Canarias queda constituida por la muestra diseñada por el INE y una muestra complementaria diseñada por el ISTAC. En este nuevo acuerdo la captura de datos de la ampliación muestral es realizada por el ISTAC, que a su vez es el responsable de la recogida de información de vuelos nacionales que queda fuera del acuerdo.

1.1.1. Objetivos

Los principales objetivos de esta operación son:

1. Estimar el número turistas procedentes del extranjero entrados en cada una de las islas, según sus características.
2. Estimar el número turistas procedentes del resto del territorio nacional entrados en cada una de las islas, según sus características.
3. Estimar el número de excursionistas entrados en cada una de las islas.

1.1.2. Método de selección de vuelos

El tipo de muestreo utilizado es bietápico estratificado en las unidades de primera etapa. Esto quiere decir que para cada aeropuerto en la primera etapa se seleccionan vuelos nacionales o internacionales, estratificados según país de destino y tipo de vuelo (regular o chárter). La selección muestral de vuelos a encuestar se realiza con un mes de antelación a la ejecución del trabajo de campo. Dicha selección es automática y guiada por un algoritmo de optimización que analiza todas las posibles combinaciones de días y vuelos, obteniendo como resultado una muestra lo más eficiente posible en función de los recursos humanos disponibles para la ejecución de los trabajos de campo.

En la segunda etapa se seleccionan pasajeros, intentando captar información de todos los pasajeros, en las salas de espera para proceder al embarque. Si bien el método de selección utilizado no permite la determinación de la probabilidad de selección de cada unidad informante, el procedimiento intenta acercarse a la equiprobabilidad de selección de cada turista en cada estrato.

La muestra final que se usa en FRONTUR-Canarias, tal como indicamos anteriormente, se compone de los vuelos seleccionados por el INE y la selección complementaria que realiza el Instituto Canario de Estadística ISTAC para cumplir con los objetivos asociados a FRONTUR-Canarias.

1.1.3. Marco de selección

En el marco de selección de vuelos se utiliza el fichero GESLOT proporcionado por la empresa Aeropuertos Españoles y Navegación Aérea (AENA) de su Sistema de Gestión Aeroportuaria.

GESLOT es la herramienta utilizada por AENA para la gestión de los slots aeroportuarios, con el fin de optimizar la utilización de las capacidades disponibles en los aeropuertos de soberanía española.

Un slot aeroportuario es el permiso dado por un coordinador para utilizar toda la infraestructura aeroportuaria necesaria con fines de aterrizaje o despegue en una fecha y hora determinadas y asignadas por un coordinador de conformidad con dicho Reglamento, para la prestación de un servicio aéreo en un aeropuerto coordinado.

Para el tratamiento de los slots aeroportuarios, se divide el año en dos temporadas: temporada de verano y temporada de invierno. Se denomina temporada de verano al período de tiempo comprendido entre el último domingo de marzo y el sábado anterior al último domingo de octubre. Se denomina temporada de invierno al periodo de tiempo comprendido entre el último domingo de octubre y el sábado anterior al último domingo de marzo.

El marco de selección se estratifica según los siguientes criterios:

VUELOS INTERNACIONALES

a Aeropuerto de encuestación
b Tipo de vuelo: regular/charter
c País de destino del vuelo

VUELOS NACIONALES

a Aeropuerto de encuestación
b Tipo de vuelo: regular/charter
c Aeropuerto nacional de destino del vuelo

1.1.4. Tamaños muestrales

Para cada uno de los estratos muestrales, resultado de la combinación de aeropuerto de origen, país de destino y tipo de vuelo (regular o chárter), se definen tamaños mínimos de representación en la muestra combinada. La tabla siguiente indica los mínimos necesarios para representar a un estrato.

N^o pasajeros	N^o min. encuestas
$N < 60$	20
60-999	30
1.000-9.999	200
10.000-24.999	400
25.000-39.999	500
40.000-59.000	600
$\geq$60.000	700

Tabla 1.1. Tabla de tamaño muestral mínimo

1.2. Descripción del problema

En esta sección vamos a explicar el procedimiento del problema planteado que queremos resolver. Nuestro principal objetivo es maximizar el número de encuestas a realizar teniendo en cuenta varias restricciones.

1.2.1. Restricciones

Tenemos numerosas limitaciones en el problema que deben cumplirse y son las siguientes:

1. La jornada laboral de los encuestadores es de 8 horas como máximo.
2. No se pueden encuestar los vuelos seleccionados por el INE.
3. El número de encuestas está restringido a la tabla de tamaño muestral mínimo (1.1) .
4. En cada aeropuerto no puede haber más de dos encuestadores en el mismo día.
5. Las encuestas se realizan durante 7 días consecutivos al mes.
6. La ocupación de los aviones es del 80 % del total.
7. Se tiene una tasa de encuesta del 50 % de los pasajeros de cada vuelo seleccionado.
8. Cada encuesta se hace en un tiempo limitado de 30 segundos y debe finalizar cinco minutos antes de la salida del vuelo. Tomamos X = "N° de asientos de un avión".
   Por tanto, para calcular el tiempo que el entrevistador debe comenzar a realizar las encuestas antes de la salida del vuelo, realizamos la siguiente operación:

$$80\,\% * 50\,\% * X * 30s + 5min$$

- $80\,\% \equiv$ Porcentaje de ocupación de las aeronaves.
- $50\,\% \equiv$ Porcentaje del número de pasajeros a encuestar.
- 30 segundos es el tiempo que se tarda en realizar una encuesta.
- 5 minutos antes de la hora de salida del vuelo es el tiempo límite para hacer la última entrevista.

En el caso de que dos encuestadores estén en un mismo vuelo, se precisarían de la mitad de tiempo.

9. Los aeropuertos canarios donde se realizan las encuestas son:

Código	***Aeropuerto***
TFS	***Tenerife Sur***
TFN	***Tenerife Norte***
LPA	***Gran Canaria***
FUE	***Fuerteventura***
ACE	***Lanzarote***
SPC	***La Palma***

Tabla 1.2. Tabla de Aeropuertos Canarios

1.2.2. Modelo matemático

Primeramente seleccionamos un aeropuerto canario y la semana que se desea encuestar. A continuación, formulamos el problema.

Tenemos los siguientes parámetros:

- $V \equiv$ Número de vuelos que salen de un aeropuerto canario en una semana concreta. Pueden ser nacionales o internacionales.
- $K \equiv$ Número de encuestadores en un aeropuerto canario.
- $Num\_min\_V \equiv$ Número mínimo de encuestas a realizar según el número de asientos del vuelo. Ver tabla 1.1.
- $Num\_País \equiv$ Número de países a los que se dirigen los vuelos con origen un aeropuerto canario.
- $J \equiv$ Jornada laboral en horas = 8 horas.
- $T \equiv$ Tiempo en minutos que se tarda en realizar una encuesta.

- $Num\_Vue\_INE \equiv$ Número de vuelos que encuesta el INE durante dicha semana en un aeropuerto canario.
- $Ocupación \equiv$ Porcentaje de ocupación sobre el número de asientos de un avión.
- $Entrevistas \equiv$ Porcentaje de encuestas que se hacen sobre el número total de pasajeros.
- $Entre = \{1, ..., K\}$.
- $Vuelo = \{1, ..., V\}$.
- $País = \{1, ..., Num\_País\}$.
- $VueloINE = \{1, ..., Nm\_Vue\_INE\}$.
- $país_i \equiv$ País al que se dirige el vuelo i, $i \in Vuelo$.
- $hora_i \equiv$ Hora a la que despega el vuelo i, $i \in Vuelo$.
- $num\_asientos_i \equiv$ Número de asientos que tiene el avión en el vuelo $i, i \in Vuelo$.
- $día_i \equiv$ Día en el que se realiza el vuelo $i, i \in v$.
- $encuestas_i \equiv$ Número de encuestas que se hacen al vuelo $i, i \in Vuelo$.
- $consumo_i \equiv$ Tiempo que se tarda en encuestar un vuelo i si hubiera un único entrevistador en el aeropuerto canario, $i \in Vuelo$.
- $país\_INE_t \equiv$ País al que se dirige el vuelo t encuestado por el INE, $t \in VueloINE$.
- $num\_asientos\_INE_t \equiv$ Número de asientos del vuelo t entrevistado por el INE, $t \in VueloINE$.
- $hora\_INE_t \equiv$ Hora de salida del vuelo t encuestado por el INE, $t \in VueloINE$.
- $día\_INE_t \equiv$ Día en el que se realiza el vuelo t encuestado por el INE, $t \in VueloINE$.
- $encuestas\_INE_j \equiv$ Número de encuestas hechas por el INE a cada país $j, j \in País$.
-

$$L_i = \begin{cases} 1, \text{ si el vuelo } i \text{ no es encuestado por el INE, } i \in Vuelo. \\ 0, \text{ si el vuelo } i \text{ es encuestado por el INE, } i \in Vuelo. \end{cases} \tag{1.1}$$

Definimos las variables:

-

$$x_{i,k} = \begin{cases} 1, \text{ si el encuestador } k \text{ debe trabajar sobre el vuelo } i. \\ 0, \text{ en otro caso.} \end{cases} \tag{1.2}$$

$\forall i \in Vuelo$ y para cada encuestador $k \in Entre$.

-

$$y_i = \begin{cases} 1, \text{ si se encuesta el vuelo } i. \\ 0, \text{ en otro caso.} \end{cases} \tag{1.3}$$

$\forall$ i $\in$ Vuelo.

El problema a resolver es el siguiente:

$$\max \sum_i encuestas_i * y_i \text{ para todo } i \in Vuelo. \text{ sujeto a:}$$

1. $y_i \leq L_i, \forall i \in Vuelo.$
2. $\sum_k x_{i,k} \leq 2 * y_i, \forall i \in Vuelo, k \in Entre.$
3. $\sum_k x_{i,k} \geq y_i, \forall i \in Vuelo, k \in Entre.$
4. $Parai, j \in Vuelo$ donde $i \neq j$, $día_i = día_j$ y $hora_j \geq hora_i$ podemos distinguir cuatro casos:
   a) Si $hora_j - hora_i < \frac{consumo_j}{2}, entonces$:
   $$x_{i,k} + x_{j,k} \leq 1,$$
   $\forall k \in Entre.$
   b) Si $hora_j - hora_i > J * 60 - \frac{consumo_i}{2}, entonces$:
   $$x_{i,k} + x_{j,k} \leq 1,$$
   $\forall k \in Entre.$
   c) Si $hora_j - hora_i < consumo_j, entonces$:
   $$x_{i,k} + x_{j,k} \leq 1 + \sum_l x_{j,l},$$
   $\forall k, l \in Entre$ y $l \neq k.$
   d) Si $hora_j - hora_i > j * 60 - consumo_i, entonces$:
   $$x_{i,k} + x_{j,k} \leq 1 + \sum_l x_{i,l},$$
   $\forall k, l \in Entre$ y $l \neq k.$
5. $\sum_i encuestas_i * y_i \geq Num\_Min\_V_j, \forall i \in Vuelo$, $j \in País$ y $j = país_i.$
6. $x_i, k \in \{0, 1\}, \forall i \in Vuelo$ y $\forall k \in Entre.$
7. $y_i \in \{0, 1\}, \forall i \in Vuelo.$

2

Propuesta de un algoritmo en Programación Lineal Entera.

En esta sección vamos a hablar de varios tipos de problemas de programación lineal entera, los cuales tienen relación con nuestro problema en particular.

Tratamos el siguiente problema:

Hay n trabajos con tiempos de procesamiento dados y un intervalo para la hora de inicio de cada trabajo. Cada trabajo debe procesarse, sin interrupción, en cualquier conjunto ilimitado de máquinas idénticas. Una máquina puede procesar cualquier trabajo, pero no más de un trabajo en cualquier momento. Queremos encontrar los tiempos de inicio de cada trabajo de manera que la cantidad de máquinas necesarias para procesar todos los trabajos sea mínima. Además, se debe encontrar la asignación de trabajos a cada máquina (el cronograma de la máquina). Si cada trabajo tiene un tiempo de inicio fijo el problema es un caso especial del problema de Dilworth. Lo llamamos el problema fijo del horario de trabajo (FSP). Cuando los tiempos de inicio del trabajo son variables, el problema se conoce como problema de planificación variable (VSP), para el cual no existe un procedimiento conocido de solución exacta. Como un problema de programación mínima de recursos, otras aplicaciones son evidentes, por ejemplo, encontrar la flota mínima de aviones y el horario de vuelo de cada miembro, o encontrar la flota mínima de buques cisterna para cumplir con un horario fijo de transporte.

2.1. El problema de horario de trabajo fijo (FSP)

En el problema FSP tenemos que encontrar:

1. La cantidad mínima de máquinas necesarias para procesar todos los trabajos en un horario de tiempo fijo.

2. Cadenas mutuamente excluyentes y exhaustivas.

Nuestro horario fijo maneja el caso de los tiempos de configuración r_{ij} cuando $r_{ij} = r_i, \forall j$. Para este caso c_i se puede extender por r_i unidades.

Algoritmo de construcción de cadenas para FSP

1. Reemplazar todos los trabajos necesariamente unidos por un solo trabajo.
2. Tomar cualquier punto de inicio arbitrario $s \in R_0$ y marcar el trabajo al que pertenece. Encontrar el final de este trabajo $c \in R_i$. Si $i < k$, proceder marcando un nuevo trabajo con una hora de inicio $s' \in R_I, s' \geq c$. Continuar hasta $s' \in R_k$. Si $R_i = R_k$, unir a la cadena cualquier secuencia arbitraria de trabajos en la región R_k y marcarlos.
3. Eliminar todas las tareas marcadas del programa y asignarlas a una sola cadena. Considerar los trabajos restantes como un nuevo horario fijo. Si no está vacío, regresar a 1. De lo contrario, detenerse.

2.1.1. Problemas NP-hard

Definición 1: *Un problema NP es un conjunto de problemas que pueden ser resueltos en tiempo polinómico.*

Definición 2: *Un problema P es NP-hard cuando para cualquier problema L en NP hay una reducción del tiempo polinomio de L a P.*

2.1.2. Problema de programación del conductor de guagua (BDSP)

Consiste en encontrar un conjunto de deberes del conductor que cubra el cronograma a un costo mínimo, al tiempo que satisface un conjunto de restricciones establecidas por el contrato sindical y las reglamentaciones de la compañía. Define un punto de alivio como un punto a lo largo de una ruta donde un conductor puede salir y tomar el autobús, y un tiempo de alivio como un momento durante el día cuando un autobús pasa un punto de alivio. Un incremento de trabajo es entonces una parte del trabajo entre dos tiempos de alivio adyacentes del mismo autobús. Una restricción común es un límite superior en el tiempo total entre el inicio y el final del trabajo de cualquier conductor. Teniendo en cuenta esta limitación, se tiene el BDSP simplificado para el cual un algoritmo

exacto puede ser de interés práctico para proporcionar límites más bajos para el problema general o para resolver subproblemas que surgen cuando BDSP se resuelve heurísticamente.

Consideraremos incrementos de trabajo y controladores en lugar de tareas y procesadores. Supondremos, sin pérdida de generalidad, que:

a) Todos los datos numéricos son enteros positivos.
b) $r_j < d_j \leq r_j + s$ para $j = 1, ..., n$.
c) El controlador que ejecuta un incremento de trabajo j funciona en un intervalo de tiempo $(r_j, d_j]$.
d) $r_1 \leq r_2 \leq ... \leq r_n$.

Por tanto, tenemos el siguiente problema (P):

$$\begin{gathered}\text{Min z} = \textstyle\sum_{i=1}^{k} m\ y_i \\ \text{sujeto a:} \\ x_{i,j} \leq y_i,\ i = 1,\ ...,\ m;\ j = 1,\ ...,\ n. \\ \textstyle\sum_{i=1}^{m} x_{i,j} = 1,\ j = 1,...,\ n. \\ x_{i,j} + x_{i,k} \leq 1,\ i = 1,..,\ m;\ j = 1,...,n,\ k \in I_j. \\ x_{i,j} \in \{0, 1\},\ i = 1,..,\ m;\ j = 1,...,n. \\ y_i \in \{0, 1\},\ i = 1,\ ...,\ m.\end{gathered}$$

Este problema es NP-hard, es decir, no admite soluciones exactas eficientes.

Algoritmo para el problema BDSP

Describimos un algoritmo de ramificación y unión para la solución exacta del problema P. El algoritmo incorpora los límites inferiores y los criterios de dominancia descritos. Además, aplicamos una fase de preprocesamiento para reducir el tamaño del problema a través de las siguientes propiedades inmediatas:

1. Si un incremento de trabajo j es incompatible con todas los demás, entonces puede asignarse, como una obligación única, a un controlador y borrarse de la lista del incremento de trabajo.
2. Si un incremento de trabajo j es incompatible con todos los demás, excepto uno (por ejemplo k), entonces j y k pueden asignarse a un controlador y borrarse de la lista de incremento de trabajo.

Aplicando estas propiedades a todos los incrementos de trabajo y repitiendo lo mismo para los restantes, y así sucesivamente, hasta que no se pueda obtener más reducción, requeriría $O(n^3)$ tiempo. Sin embargo, el mismo resultado se puede lograr en el tiempo $O(n^2)$, de la siguiente manera:

Procedimiento de Reducción

1. $H := \{1, ..., n\}$;
2. for each $h \in H$ do compute g_h=n° of iow's compatible with h.
3. exit := false;
   repeat
4. find $g_j = min\{g_h \in H\}$
5. if $g_j > 1$ then exit := true else
6. if $g_j = 0$ then (comment property (i))
   begin
7. assign j to a new driver;
8. $H := H - \{j\}$
   end
   else (comment property (ii))
   begin

9. find (the unique) $k \in H$ compatible with j;
10. assign j and k to a new driver;
11. $H := H - \{j, k\}$;
12. for each $h \in H$ compatible with k do
    $g_h := g_h - 1$
    end

13. until exit.

Después de la reducción, se realiza un algoritmo de bifurcación de primer grado en profundidad sobre el de incremento de trabajo en H. Los incrementos de trabajos están ordenados de modo que $r_j \leq r_j + 1 \forall j$. El algoritmo comienza a introducir el controlador 1 y asignarle al incremento de trabajo 1.

En cada nodo u del árbol de decisión, supongamos que $i(u)$ sea el último controlador introducido: los deberes de los controladores 1, ..., $i(u)$ - 1 son máximos, en el sentido de que no se les puede asignar más incrementos de trabajo. Sea $F(u) = (f(u), ..., l(u))$ la cadena factible asignada actualmente a $i(u)$, entonces:
$A(u) = j : j$ is unassigned, $r_j \geq d_l(u)$ and $d_j \leq r_f(u) + s$
es el conjunto de todos los incrementos de trabajo que se podrían asignar a $i(u)$. El algoritmo considera candidatos para el siguiente incremento de trabajo de $i(u)$ aquellos en
$S(u)$ = $A(u)$ - $k \in A(u)$:k está dominado por $j \in A(u), j < k$, con respecto a $F(u)$.

El nodo u genera el siguiente nodo hijo al asignar a $i(u)$ el siguiente incremento de trabajo en $S(u)$.

Cuando $A(u) = \phi$, el deber del conductor $i(u)$ es máximo. Solo en tal situación probamos si su nodo puede ser explorado, es decir, si puede ocurrir un retroceso. Si este no es el caso, se genera un nodo hijo al introducir un nuevo controlador $i(u) + 1$ y asignarle el primer enlace no asignado. Para establecer si el nodo u puede ser entendido tomamos z como el valor de la mejor solución hasta el momento y definimos

* $J(u) = j$: incremento de trabajo j está actualmente sin asignar;
* $p(u) = |i$:en la mejor solución hasta el momento, realizo al menos un incremento de trabajo $j \in H - J(u)|$;
* $L(u) = i(u) +$ (valor de L_2 calculado para el subproblema determinado por $J(u)$);
* $\Delta t(u) =$ max d_j: $j \in J(u)$ - min $r_j : j \in J(u)$.

Entonces el retroceso puede ocurrir claramente en las siguientes situaciones:

1. i(u) $\geq$ p(u)
2. L(u) $\geq$ z
3. $\Delta t(u) \leq s$

Usando la notación anterior, podemos delinear una posible implementación del algoritmo de la siguiente manera.

ALGORITMO

1. apply procedure REDUCTION and remove from the instance the assigned iow's;
   let 1, ..., n be the iow's in the reduced instance;
2. compute t_α, t_β, n' and n'';
   if $n' > n''$ then replace the instance with its reversed-time version;
3. sort the iow's according to increasing release time;
4. $i := 0$; $z := \infty$;
   branch (comment initialize a new driver)
5. let u be the current node of the branch-decision tree;
6. if $i \geq p(u)$ then go to backtrack;
7. if $\Delta t(u) \leq s$ then
   begin
   solve the instance of P_2 determined by $J(u)$ and let τ be the required number of drivers;
   if $i+\tau < z$ then $z := i+\tau$ and update the optimal solution:
   go to backtrack
   end;
8. $LB_i := i+$ (value of L_2 computed over the inassigned iow's);
9. if $LB_i \geq z$ then go to backtrack;
10. $i := i + 1$ and generate a new node u by assigning to i the first unassigned iow;

ALGORITMO

11. let j be the last iow, if any, emoved (by backtracking) from the duty of i;
12. $S := \{l \in A(u) : l > j\}$
13. while $S \neq \phi$ do
    begin
14. let k be the first iow in S and F the current duty of i;
15. if Theorem 4.1 applies to some unassigned iow j' then go to backtrack;
16. if j dominates k with respect to F then $S := S - \{k\}$
    else assign k to i and set $S := S - (\{k\} \cup \{l \in S : loverlapsk\})$
    end;
17. go to branch; backtrack:

18. while $i > 0$ and $LB_{i-1} = z$ do
    begin
19. deassign all iow's assigned to i;
20. $i := i - 1$
    end;
21. while $i > 0$ do
    begin
22. deassign the last iow assigned to i;
23. if at least one iow is currently assigned to i
    then go to forward
    else $i := i - 1$
    end
    end

2.2. El problema del horario variable (VSP)

Cuando se inician intervalos de tiempo $[a_i, b_i]$, $i = 1, ..., n$ se dan, entonces existe una colección $S' = \{S|s_i \in [a_i, b_i]\}$ de todos los horarios fijos posibles. Se puede definir VSP como:
Dado a_i, b_i, t_i, encontrar:

1. Un trabajo fijo programa S^* tal que H $(S^*) = \min H(S)$ $S \in S'$.
2. Un conjunto de H (S^*) cadenas correspondientes a S^*.

Solución aproximada a VSP

Si se otorgan tolerancias en los horarios de cada trabajo, el problema de encontrar el horario fijo que dé como resultado el número mínimo de máquinas para procesar todos los trabajos es la primera parte de VSP. Dada una solución a esta parte del problema, los métodos de descomposición de la cadena se pueden usar para resolver la segunda parte de VSP.

Se discuten dos procedimientos heurísticos para encontrar una solución aproximada a la primera parte de VSP. El primero usa algunas ideas probabilísticas basadas en entropía y 'suavizado informativo', mientras que el segundo emplea la representación de función escalonada como guía para desplazar el trabajo de las áreas máximas funcionales.

Minimización local en áreas máximas

Dado un cronograma fijo, se tiene en cuenta que los puntos finales del área máxima de $f_s(t)$ están asociados con uno o dos trabajos. Se intenta reducir el valor máximo de la función funcional observando todos los cambios posibles de estos trabajos. Este proceso continúa hasta que las áreas máximas no se pueden reducir. El procedimiento se puede iniciar diseminando el primer horario tentativo de acuerdo con $s_i = \frac{b_i + a_i}{2}$ para cada trabajo i. Se construye el $f_s(t)$ funcional y se identifican las regiones donde toma los valores máximos. En general, los puntos finales de estas regiones están formados por dos trabajos superpuestos o un solo trabajo.

Presentamos el siguiente principio. Tratamos de reducir la cantidad máxima funcional en todos los turnos posibles de cualquier trabajo o par de trabajos que formen la región máxima. Si la región máxima se reduce en uno de estos cambios y no se aumenta o introduce ninguna otra región máxima, se realiza el cambio de trabajo. Esto forma un nuevo horario. El proceso continúa hasta que no hay mejoras posibles.

La experiencia práctica en la programación indica que este procedimiento se puede realizar manualmente para problemas muy grandes y logra la mayor parte de las posibles reducciones de la máquina.

Experiencia computacional

El propósito de los experimentos fue investigar la eficiencia del algoritmo de entropía. Esta eficiencia se calcula como la relación entre el número mínimo de máquinas obtenidas por entropía y el número óptimo de máquinas. Una relación de 1.00 implica que la solución del algoritmo de entropía heurística es óptima.

Ambos algoritmos se probaron en 40 problemas aleatorios que varían en tamaño de 5 a 100 trabajos y de 30 a 100 períodos. Los problemas se generaron al seleccionar tres parámetros: n, el número de trabajos; T, la longitud máxima del horario; y τ, un número entero que restringe las tolerancias de tiempo de inicio del trabajo. el primer paso en la generación de cronogramas consiste en las siguientes operaciones:

1. Tome dos realizaciones independientes α_1, α_2 de la población $\alpha \sim U(0, T)$.
2. Establece $s_i = min(\alpha_1, \alpha_2)$ y $c_i = max(\alpha_1, \alpha_2)$.
3. Tome un entero aleatorio β de la población $\beta \sim U(1, \tau + 1)$.
4. Establezca $\Delta_i^1 = min(\beta, s_i)$; $\Delta_i^2 = min(T - c_i, \beta)$; $\Delta_i^3 = min(\Delta_i^1, \Delta_i^2)$.
5. Determine los parámetros del trabajo ith $= a_i = s_i - \Delta_i, b_i = s_i + \Delta_i, t_i = c_i - s_i$.

Este método produce horarios con una distribución uniforme de las longitudes de trabajo. Esto también es cierto para las tolerancias de tiempo de inicio del trabajo, con la excepción de los casos en que el trabajo está demasiado cerca de los puntos 0 y T. En estos casos, las tolerancias, como se ve en las fórmulas anteriores, se truncan para forzar el inicio más temprano del trabajo y los últimos tiempos de finalización para que caigan dentro del intervalo [0, T].

3

Experimentación sobre datos reales proporcionados por el ISTAC.

El ISTAC nos proporciona ficheros Excel sobre los datos que poseen para realizar las encuestas en los distintos aeropuertos canarios. En primer lugar, implementamos un código en C para crear un fichero en el que se muestran los vuelos de un aeropuerto canario en concreto durante toda una semana y, posteriormente, desarrollamos otro código en CPLEX, basado en lenguaje C, donde utilizamos el algoritmo para obtener la solución óptima al problema.

3.1. Implementación en C

Tenemos un fichero Excel el cual tiene las siguientes columnas:

- Destino : Lugar de destino.
- Código : Código del vuelo.
- Día_semana : Días de la semana (L, M, X, J, V, S, D).
- Opera_desde : Fecha en la que el vuelo empieza a operar.
- Opera_hasta : Fecha en la que el vuelo termina de operar.
- Hora_salida : Hora de salida del vuelo.
- Aeronave : Tipo de avión.
- Num_vuelo : Número del vuelo.
- País : País de destino.
- Escala : Indica el lugar donde se produce escala.
- Origen : Aeropuerto canario de origen.
- Num_asientos : Número de asientos.

		Destino	Codigo	Dia_semana	Opera_desd	Opera_hasta	Hora_Salida	Aeronave	Num_vuelo	Pais	Escala	Origen
W17	Destino	AMSTERDAM/SCHIPHOL	AMS	D	05/11/2017	18/03/2018	10:10	73H	TRA5686	HOLANDA		ACE
W17	Destino	AMSTERDAM/SCHIPHOL	AMS	S	04/11/2017	16/12/2017	10:25	73H	TRA5684	HOLANDA		ACE
W17	Destino	AMSTERDAM/SCHIPHOL	AMS	M	07/11/2017	20/03/2018	10:25	73H	TFL530	HOLANDA		ACE
W17	Destino	AMSTERDAM/SCHIPHOL	AMS	J	02/11/2017	22/03/2018	10:45	73H	TRA5686	HOLANDA		ACE
W17	Destino	AMSTERDAM/SCHIPHOL	AMS	M	07/11/2017	21/11/2017	10:45	73H	TRA5686	HOLANDA		ACE
W17	Destino	AMSTERDAM/SCHIPHOL	AMS	M S	31/10/2017	30/01/2018	12:00	320	EZY7948	HOLANDA		ACE
W17	Destino	AMSTERDAM/SCHIPHOL	AMS	J	02/11/2017	04/01/2018	16:30	73H	TFL532	HOLANDA		ACE
W17	Destino	AMSTERDAM/SCHIPHOL	AMS	S	04/11/2017	25/11/2017	17:45	73H	TRA6186	HOLANDA		ACE
W17	Destino	AMSTERDAM/SCHIPHOL	AMS	D	05/11/2017	19/11/2017	18:00	788	TFL534	HOLANDA	FUE	ACE
W17	Destino	ASTURIAS	OVD	S	04/11/2017	24/03/2018	14:25	320	VLG3138	ESPAÑA		ACE
W17	Destino	BARCELONA-EL PRAT	BCN	L_VSD	29/10/2017	24/03/2018	15:45	321	VLG2473	ESPAÑA		ACE
W17	Destino	BARCELONA-EL PRAT	BCN	MX	31/10/2017	03/01/2018	15:45	321	VLG2473	ESPAÑA		ACE
W17	Destino	BARCELONA-EL PRAT	BCN	M	31/10/2017	19/12/2017	16:50	73H	RYR6373	ESPAÑA		ACE
W17	Destino	BARCELONA-EL PRAT	BCN	J	02/11/2017	22/03/2018	16:50	73H	RYR6373	ESPAÑA		ACE
W17	Destino	BARCELONA-EL PRAT	BCN	S	04/11/2017	04/11/2017	16:50	73H	RYR6373	ESPAÑA		ACE
W17	Destino	BASEL/MULHOUSE EUROAIRPORT CH	BSL	X S	01/11/2017	24/03/2018	17:25	320	EZS1250	SUIZA		ACE
W17	Destino	BELFAST / INTERNACIONAL	BFS	L V	30/10/2017	15/12/2017	7:20	73H	RYR3498	REINO UNIDO		ACE
W17	Destino	BELFAST / INTERNACIONAL	BFS	J	02/11/2017	22/03/2018	13:45	32B	TCX1229	REINO UNIDO		ACE
W17	Destino	BELFAST / INTERNACIONAL	BFS	S	04/11/2017	30/12/2017	14:00	73H	EXS310	REINO UNIDO		ACE
W17	Destino	BELFAST / INTERNACIONAL	BFS	X	01/11/2017	21/03/2018	15:00	73H	EXS310	REINO UNIDO		ACE
W17	Destino	BELFAST / INTERNACIONAL	BFS	D	29/10/2017	19/11/2017	15:50	32B	TCX1229	REINO UNIDO		ACE

Figura 3.1. Fichero de datos del ISTAC

Nuestro objetivo es crear un fichero que contenga las columnas:

- Destino
- Hora_salida
- Día_semana
- Código
- Num_asientos

Para ello, guardamos el fichero *Aeropuertos Canarios* en la extensión '.csv' ya que resulta más cómodo a la hora de trabajar con él. Luego, implementamos un código en C donde se seleccionan las cinco columnas que nos interesa y se copian en un fichero nuevo que creamos.
En dicho código lo que hacemos es introducir, en primer lugar, el nombre del fichero que queremos leer y luego el nombre del archivo nuevo en el que se van a copiar los datos que nos interesa. Posteriormente, pedimos por pantalla el aeropuerto origen y la fecha que deseamos estudiar. Ahora el programa recorre el fichero y con la función "strtok"dividimos un string en ";", lo guarda en una variable y copiamos las deseadas. También tenemos que separar las fechas ya que se introducen en el formato 'DD/MM/AAAA' y a nosotros lo que nos interesa es toda la semana. Además, en el fichero que nos entrega el ISTAC aparecen las variables 'Opera_desde', 'Opera_hasta' y 'Día_semana'. En esta última pueden aparecer varios días a la semana, con lo cual, tenemos que averiguar qué día es exactamente en esa semana. Un claro ejemplo de esto puede ser el caso siguiente:

- Día_semana = M S, esto es martes y sábado.
- Opera_desde = 31/10/2017.
- Opera_hasta = 30/01/2018.
- Semana deseada comienza el 02/01/2018.

Entonces, el programa devuelve los días 02/01/2018 y 06/01/2018 que pertenecen al mismo vuelo.

A continuación, podemos observar el código en C:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctime>


typedef struct{
int dia;
int mes;
int ano;
} fecha;

int main(){

FILE *fentrada, *fsalida;
char nombre[80], aux[200], aux1[200], aux2[200], aux3[200],
 origen[5], origenDeseado[10];
char destino[50], hora_salida[50], codigo[50], dia_semana[10],
 opera_desde[15], opera_hasta[15], asiento[15];
fecha dia;
int i;

printf("Introduce el nombre del fichero a leer con la
información del ISTAC: ");
scanf("%s", nombre);
fentrada = fopen(nombre, "rt");
if (fentrada == NULL) {
printf("No existe el fichero %s",nombre);
return 1;
}

printf("Introduce el nombre del fichero a generar con los vuelos: ");
scanf("%s", nombre);
fsalida = fopen(nombre, "wt");
```

```
if (fsalida == NULL) {
printf("No es posible crear el fichero %s",nombre);
return 2;
}

fgets(aux, 200, fentrada);
char* item;

struct tm desde;
struct tm hasta;
struct tm semana;

printf("Introduzcael aeropuerto origen");
scanf("%s", origenDeseado);
printf("Introduzca la semana que desee con el siguiente
formato DD/MM/AAAA:");
scanf("%d/%d/%d", &semana.tm_mday, &semana.tm_mon, &semana.tm_year);
semana.tm_mon -= 1;
semana.tm_year -= 1900;

do {

fgets(aux, 200, fentrada);

if (aux != NULL){
item = strtok(aux, ";"); //La función strtok divide
un string en ; y devuelve esa parte a la variable item

item = strtok (NULL, ";");
item = strtok (NULL, ";");
strcpy(destino, item);

item = strtok (NULL, ";");
strcpy(codigo, item);

item = strtok (NULL, ";");
strcpy(dia_semana, item);

item = strtok (NULL, ";");
strcpy(opera_desde, item);

item = strtok (NULL, ";");
strcpy(opera_hasta, item);
```

```
item = strtok (NULL, ";");
strcpy(hora_salida, item);

for (i = 0; i < 4; i++){
item = strtok (NULL, ";");
}
strcpy(origen, item);

item = strtok (NULL, ";\n");
strcpy(asiento, item);
int numero = atoi(asiento);

/*Separar fechas*/
item = strtok (opera_desde, "/");
desde.tm_mday = atoi(item);//Cambiar cadena a número
item = strtok (NULL, "/");
desde.tm_mon = atoi(item)-1;
item = strtok (NULL, "/");
desde.tm_year = atoi(item)-1900; //Por la estructura

item = strtok (opera_hasta, "/");
hasta.tm_mday = atoi(item);
item = strtok (NULL, "/");
hasta.tm_mon = atoi(item)-1;
item = strtok (NULL, "/");
hasta.tm_year = atoi(item)-1900;

time_t tiempo_desde = mktime(&desde); //Paso de struct a time_t
time_t tiempo_hasta = mktime(&hasta);
time_t tiempo_semana = mktime(&semana);
time_t tiempo_final;
struct tm* final; //Puntero

if(!strcmp(origen, origenDeseado))

if(difftime(tiempo_hasta, tiempo_semana) > 0 &&
difftime(tiempo_semana,tiempo_desde) > 0){
//Esto siginifica que está en la semana deseada
printf("\n");
for (i=0;i<7;i++){
struct tm semanaP = semana;
if(dia_semana[i] != ' '){
```

```
char dia_final[10];
if (dia_semana[i] == 'L'){ //Si el día de la semana es lunes
tiempo_final = mktime(&semanaP);
tiempo_final = tiempo_final + 24*3600*(6);
final = gmtime(&tiempo_final);
strftime (dia_final, 80, "%d/%m/%Y", final);
}
else{
tiempo_final = mktime(&semanaP);
tiempo_final = tiempo_final + 24*3600*(i-1);
final = gmtime(&tiempo_final);
strftime (dia_final, 80, "%d/%m/%Y", final);
}
fprintf(fsalida, "%s;%s;%s;%d;%s\n", destino, hora_salida,
  codigo, numero, dia_final);
}
}
}
}
} while (!feof(fentrada));
fclose(fsalida);
fclose(fentrada);

return 0;
}
```

En el caso de que el archivo no contenga la columna Num_asientos simplemente añadimos al código anterior las siguientes líneas con el número de asientos de cada aeronave:

```
if (strcmp(type, "319") == 0) return 156;
if (strcmp(type, "320") == 0) return 220;
if (strcmp(type, "321") == 0) return 186;
if (strcmp(type, "32A") == 0) return 220;
if (strcmp(type, "32B") == 0) return 220;
if (strcmp(type, "32S") == 0) return 220;
if (strcmp(type, "332") == 0) return 335;
if (strcmp(type, "333") == 0) return 295;
if (strcmp(type, "73C") == 0) return 100;
if (strcmp(type, "73G") == 0) return 100;
if (strcmp(type, "73H") == 0) return 100;
if (strcmp(type, "73J") == 0) return 100;
if (strcmp(type, "73W") == 0) return 100;
```

```
if (strcmp(type, "7S8") == 0) return 189;
if (strcmp(type, "75T") == 0) return 189;
if (strcmp(type, "75W") == 0) return 189;
if (strcmp(type, "738") == 0) return 189;
if (strcmp(type, "788") == 0) return 210;
if (strcmp(type, "7M8") == 0) return 162;
if (strcmp(type, "737") == 0) return 100;
if (strcmp(type, "734") == 0) return 189;
if (strcmp(type, "752") == 0) return 239;
if (strcmp(type, "767") == 0) return 278;
if (strcmp(type, "76W") == 0) return 181;
if (strcmp(type, "AT7") == 0) return 72;
if (strcmp(type, "ATR") == 0) return 72;
if (strcmp(type, "BE4") == 0) return 100;
if (strcmp(type, "CRK") == 0) return 100;
if (strcmp(type, "E95") == 0) return 100;
if (strcmp(type, "EP3") == 0) return 100;
if (strcmp(type, "DH4") == 0) return 100;
if (strcmp(type, "M83") == 0) return 100;
if (strcmp(type, "H28") == 0) return 100;
```

3.2. Optimización del problema

Para resolver el problema utilizaremos el fichero nuevo creado con el código anterior y Cplex, un software de optimización que resuelve problemas de programación entera y de programación lineal muy grandes y, además, está implementado en el lenguaje de programación C.

Utilizando otro código desarrollado por mi tutor, Juan José Salazar, que nos permite leer el fichero de datos obtenido anteriormente y usando CPLEX para resolver el modelo matemático hemos podido obtener las soluciones que solventa el problema de manera óptima. Este código genera un fichero en el cual se muestran en diferentes columnas el código de vuelo, el tipo de aeronave, la fecha del vuelo, la hora de salida, el código del aeropuerto canario de origen, el código del aeropuerto de destino, el país de destino y el entrevistador seleccionado para encuestar el vuelo, como podemos ver en el siguiente ejemplo:

vuelo	tipo	dia	hora	origen	destino	region	entrevistador
IBB501	CRK	02/01/2018	8:15	ACE	LPA	ESPAÑA	0
IBB501	CRK	02/01/2018	8:15	ACE	LPA	ESPAÑA	1
IBB453	AT7	02/01/2018	8:25	ACE	TFN	ESPAÑA	0
VLG3141	320	02/01/2018	8:45	ACE	AGP	ESPAÑA	1
IBB521	AT7	02/01/2018	8:50	ACE	LPA	ESPAÑA	0
VLG3281	320	02/01/2018	9:40	ACE	BIO	ESPAÑA	0
VLG3281	320	02/01/2018	9:40	ACE	BIO	ESPAÑA	1
NLY2431	321	02/01/2018	10:15	ACE	MUC	ALEMANIA	0

Esto se puede solucionar en general para todos los aeropuertos canarios y cualquier semana que escojamos, pero vamos a tomar el siguiente ejemplo para observar la solución que obtenemos al usar el modelo matemático. Elegimos la semana del 02/01/2018 y asumimos dos encuestadores, ocho horas máximas de trabajo cada día, una ocupación de los aviones del 80 %, una tasa de encuesta del 50 % de los pasajeros de cada vuelo seleccionado y 30 segundos para realizar cada encuesta. A continuación veremos las soluciones obtenidas de algunos aeropuertos canarios con estas condiciones:

Resultados Lanzarote

Por tanto, la salida que produce el algoritmo es la siguiente:

```
Total (root+branch&cut) =   24.97 sec. (13871.87 ticks)
Optimizing with time limit of 60 minutes
 Reading 7 days
 Reading 33 types of aircraft
 Reading 7 values in the table
 Reading 382 flights
 To      HOLANDA:   8 flights ;    832 seats ;   665.60 expected passengers ;  30 min survey
 To       ESPAÑA: 219 flights ;  15760 seats ; 12608.00 expected passengers ; 400 min survey
 To        SUIZA:   4 flights ;    704 seats ;   563.20 expected passengers ;  30 min survey
 To  REINO-UNIDO:  84 flights ;   9809 seats ;  7847.20 expected passengers ; 200 min survey
 To     ALEMANIA:  28 flights ;   4119 seats ;  3295.20 expected passengers ; 200 min survey
 To    DINAMARCA:   5 flights ;    638 seats ;   510.40 expected passengers ;  30 min survey
 To      BELGICA:   3 flights ;    432 seats ;   345.60 expected passengers ;  30 min survey
 To      HUNGRIA:   1 flights ;    176 seats ;   140.80 expected passengers ;  30 min survey
 To      IRLANDA:   8 flights ;    928 seats ;   742.40 expected passengers ;  30 min survey
 To    FINLANDIA:   1 flights ;    176 seats ;   140.80 expected passengers ;  30 min survey
 To      POLONIA:   2 flights ;    352 seats ;   281.60 expected passengers ;  30 min survey
 To       ITALIA:   6 flights ;    768 seats ;   614.40 expected passengers ;  30 min survey
 To      FRANCIA:   4 flights ;    580 seats ;   464.00 expected passengers ;  30 min survey
 To      AUSTRIA:   1 flights ;    148 seats ;   118.40 expected passengers ;  30 min survey
 To      NORUEGA:   3 flights ;    336 seats ;   268.80 expected passengers ;  30 min survey
 To       SUECIA:   5 flights ;    496 seats ;   396.80 expected passengers ;  30 min survey
```

Figura 3.2. Salida de CPLEX

A continuación, se muestra la hora de salida de cada vuelo que se debe encuestar y el tiempo, en minutos, que el encuestador debe empezar a hacer las entrevistas. En algunas ocasiones aparece '*' debido a que sobre ese mismo vuelo se encuentran trabajando dos encuestadores y, por tanto, cada encuestador trabaja la mitad de tiempo en ese vuelo.

Por tanto, la solución obtenida para el aeropuerto de Lanzarote en la semana del 02/01/2018 es la siguiente:

Worker 1:

02/01/2018 : 8:15-10 8:25-14 8:50-14 9:40-22* 10:15-18* 10:25-20 10:45-20 11:05-10* 11:20-7* 11:45-22* 12:00-7* 12:25-44 13:40-44 14:10-14 14:30-7* 14:40-10* 15:05-44 15:20-7* 15:55-44*

03/01/2018 : 14:10-14 14:30-7 15:10-22* 15:20-20 15:55-14 17:40-44 18:00-20 18:20-20 18:50-20 19:20-14 19:45-14 20:30-14 20:55-14 21:20-14*

04/01/2018 : 10:35-14 11:20-44 11:50-10 12:15-44 12:45-10* 13:10-37 13:30-10* 13:55-44 14:10-14 14:30-7* 14:50-37 15:10-10* 15:30-18* 15:40-10* 16:10-44 16:50-10* 17:00-7* 17:10-10* 17:35-22* 18:15-14*

05/01/2018 : 11:30-14 11:55-7 12:20-37 13:10-22* 13:30-37 13:55-10* 14:05-7* 14:30-44 14:50-20 15:30-10* 16:05-37 16:50-14 17:35-20 18:00-20 18:20-20 18:45-14 19:15-14*

06/01/2018 : 8:50-14 9:40-44 10:10-20 10:35-14 11:05-20 11:20-14 12:10-44 12:40-10 13:15-22* 13:30-20 14:05-14 14:30-10* 14:40-10* 15:10-44 15:30-10* 16:00-22* 16:30-37*

07/01/2018 : 8:15-20 8:50-14 9:55-44 10:40-37 11:40-44 11:55-14 12:40-44 13:30-44 13:55-7 14:05-7* 14:15-7* 14:25-10* 14:45-37 15:30-44 15:50-20*

08/01/2018 : 8:15-20 8:40-14 9:30-14 10:45-20 11:20-14 12:05-20 13:00-37 13:30-20 14:10-14 14:30-14 15:00-20 15:45-37

Worker 2:

02/01/2018 : 8:15-10 8:45-44 9:40-22* 10:15-18* 10:40-37 11:05-10* 11:20-7* 11:45-22* 12:00-7* 12:35-44 13:30-44 14:30-7* 14:40-10* 15:05-44 15:20-7* 15:45-37*

03/01/2018 : 8:15-20 8:40-14 9:40-44 10:45-20 11:20-14 11:45-20 12:40-44 13:40-44 14:05-14 14:30-7 15:10-22* 15:45-37*

04/01/2018 : 11:05-20 11:30-20 11:50-10 12:20-44 12:45-10* 13:05-20 13:30-10* 14:10-44 14:30-7* 14:50-37 15:10-10* 15:30-18* 15:40-10* 16:05-44 16:30-20 16:50-10* 17:00-7* 17:10-10* 17:35-22* 18:00-44 18:45-44*

05/01/2018 : 8:25-14 8:40-14 9:25-20 10:15-37 10:35-14 11:35-20 11:55-7 12:05-20 12:25-20 13:10-22* 13:35-44 13:55-10* 14:05-7* 14:30-44 15:30-10* 16:00-44*

06/01/2018 : 11:10-20 12:00-44 12:40-10 13:15-22* 13:40-44 14:10-14 14:30-10* 14:40-10* 15:05-44 15:30-10* 16:00-22* 16:25-44 17:10-44 17:35-14 18:40-44*

07/01/2018 : 13:40-44 13:55-7 14:05-7* 14:15-7* 14:25-10* 14:50-44 15:10-20 15:35-14 15:55-14 17:05-44 17:35-14 18:55-44 19:15-14 19:45-20 20:25-14 20:45-20*

08/01/2018 : 13:40-44 14:05-14 15:05-20 15:20-14 15:45-14 17:00-14 17:35-14 18:15-14 18:45-14 19:30-14 19:45-14 20:30-14 20:55-14

En la solución se puede ver los vuelos que deben encuestar cada entrevistador para maximizar el número de encuestas a realizar de manera óptima y teniendo en cuenta todas las restricciones. Además asumimos que el último pasajero se puede encuestar justo cinco minutos antes de la hora de salida. Esto conlleva a encuestar a 19413 pasajeros de los 382 vuelos que salen del aeropuerto de Lanzarote en la semana prefijada anteriormente.

Resultados Tenerife Norte

Ahora veremos la solución que produce el algoritmo para Tenerife Norte en la semana del 02/01/2018 asumiendo las mismas condiciones que en la solución del aeropuerto de Lanzarote. En este caso salen 535 vuelos de los cuales 529 vuelos son nacionales, es decir, tienen como destino algún lugar de España, y seis vuelos se dirigen a otros tres destinos internacionales (Marruecos, Senegal e Italia).

Worker 1:
02/01/2018 : 7:15-20 7:30-14 8:05-20 8:30-14 9:00-20 9:30-14 9:45-14 10:00-14 10:25-14 11:10-44 11:40-14 12:00-14 12:50-7 13:00-20 13:40-14 14:00-14 14:20-14 14:40-20 15:00-10*
03/01/2018 : 12:50-7* 13:50-22* 14:00-14 14:20-7* 14:30-7* 14:40-20 15:20-37 16:00-14 16:15-14 16:30-14 17:55-67 18:30-14 18:55-20 19:10-14 19:30-14 20:00-14 20:30-14
04/01/2018 : 13:00-20 13:50-44 14:20-7* 14:30-7* 14:40-20 15:20-37 16:00-14 16:15-14 16:30-14 17:55-67 18:15-14 18:30-14 18:50-20 19:10-14 19:30-14 20:00-20 20:30-14
05/01/2018 : 7:15-20 7:30-14 8:00-20 8:30-14 9:00-20 9:50-44 10:10-14 10:25-14 11:10-44 11:30-7* 11:40-7* 11:50-7* 12:00-14 12:35-7* 13:00-20 13:40-14 14:00-14 14:30-7* 14:55-44
06/01/2018 : 12:50-7* 13:00-7* 13:15-10* 13:50-44 14:20-14 14:40-20 15:20-37 16:15-14 17:55-67 18:30-14 19:00-20 19:20-14 20:00-20 20:30-14
07/01/2018 : 11:40-14 12:00-14 12:35-7* 13:05-44 13:25-7* 13:50-44 14:20-14 14:40-20 15:20-37 16:00-14 16:15-14 16:40-14 17:00-14 17:30-14 18:15-44 18:30-14 19:00-20 19:20-14
08/01/2018 : 7:10-14 7:35-20 8:00-20 8:30-14 9:00-20 9:50-44 10:05-14 10:25-14 11:10-44 11:30-14 12:00-14 13:05-44 13:50-44 14:20-7* 14:30-7* 14:55-44*

Worker 2:
02/01/2018 : 12:50-7 13:00-14 13:50-44 14:20-14 14:35-14 15:00-10* 15:20-37 16:00-14 16:15-14 16:30-14 17:55-67 18:30-14 19:00-20 19:20-14 19:50-20 20:30-14*
03/01/2018 : 7:15-20 7:35-20 8:00-20 9:00-44 9:30-20 9:45-14 10:00-14 10:25-14 11:10-44 11:35-20 12:00-14 12:50-7 13:00-14 13:50-22* 14:00-14 14:20-7* 14:30-7* 14:55-44*
04/01/2018 : 7:15-20 7:30-14 7:50-14 8:10-20 9:00-20 9:30-20 9:45-14 10:00-14 10:25-14 11:10-44 11:30-14 12:00-14 13:05-44 13:40-14 14:00-14 14:20-7 14:30-7* 14:55-44*
05/01/2018 : 10:10-14 10:50-20 11:10-20 11:30-7 11:40-7* 11:50-7* 12:00-14 12:35-7* 13:00-14 13:50-44 14:15-20 14:30-7* 14:40-20 15:00-14 15:15-14 15:30-14 16:00-14 16:15-14 16:30-14 17:55-67*
06/01/2018 : 7:15-20 7:30-14 8:00-20 9:00-20 9:50-44 10:10-14 10:25-14 11:10-44 11:30-14 12:00-14 12:50-7 13:00-7* 13:15-10* 14:10-44 14:55-44*
07/01/2018 : 7:00-44 7:35-20 8:00-20 9:00-44 9:30-20 9:45-14 10:00-14 10:25-14 11:10-44 11:30-20 12:00-20 12:35-7 12:50-14 13:25-7* 14:10-44*
08/01/2018 : 13:00-20 13:40-14 14:00-14 14:20-7 14:30-7* 14:40-20 15:00-20 15:30-20 16:00-20 16:15-14 16:30-14 17:00-20 17:25-14 18:15-44 18:30-14 18:55-20 19:10-14 19:30-14 20:00-20 20:30-20*

Resultados Tenerife Sur

Para este caso se tienen 410 vuelos con aeropuerto origen Tenerife Sur de los cuales asumimos que se tiene un 80 % de la ocupación y se estima que 39497 pasajeros saldrán de dicho aeropuerto en la semana del 2 al 8 de Enero de 2018. Por tanto, el algoritmo encuentra la solución óptima del problema y el resultado obtenido es que se deben encuestar a 21341 pasajeros.

3.3. Conclusiones

Como conclusión final, hemos observado que usando el modelo matemático explicado en el primer capítulo el problema planteado por el ISTAC se resuelve de una manera óptima. Utilizando los códigos en lenguaje C, tanto en DevC++ y CPLEX, podemos resolver cualquier planificación que necesite el ISTAC para encuestar al máximo número de pasajeros en cualquier aeropuerto canario y en la semana deseada. Siempre debemos de tener en cuenta las restricciones ya que, en caso contrario, obtendríamos un problema no factible.

Bibliografía

[1] J.J. Salazar.
Programación Matemática. Editorial Díaz de Santos, 2001.

[2] Archivo de datos Aeropuertos Canarios.
Aeropuertos_Canarios.xls

[3] ISTAC. Estadística de movimientos turísticos en fronteras de Canarias (FRONTUR-Canarias).
http://www.gobiernodecanarias.org/istac/galerias/documentos/E16028B/metodologia_FRONTUR.pdf

[4] Matteo Fischetti, Silvano Martello y Paolo Toth.
The fixed job schedule problem with spread-time constraints.

[5] Ilya Gertsbakh y Helman I. Stern.
Minimal resources for fixed and variable job schedules.

[6] Guía de aviones comerciales.
https://www.emptyleg.com/es/guia-de-aviones/guia-aviones-comerciales.

Lista de Tablas

Lista de Figuras

Algorithms for the optimal selection of flights in the survey ISTAC tourism

Zuleica Reina Segura

Facultad de Ciencias · Sección de Matemáticas
Universidad de La Laguna
alu0100773002@ull.edu.es

Abstract

The FRONTUR-Canarias survey conducted by the ISTAC aims to study tourism demand in the Canary Islands. For this, a mathematical model is needed that solves the problem of maximizing the number of surveys taking into account that surveys of all possible destinations should be included to obtain as much information as possible. In addition, we have to consider several limitations when it comes to solving the problem. To develop the algorithm that solves the problem, we implement a code in C and use the CPLEX to obtain the solutions.
FRONTUR Survey – ISTAC – Mathematical model – Algorithm.

1. Introduction

The objective of this end-of-degree project is to maximize the number of surveys that the ISTAC has to carry out considering the following restrictions:

* The work day of the interviewers is 8 hours maximum.
* You can not poll the flights selected by the INE.
* The number of surveys is restricted to the minimum sample size table.
* At each airport there can not be more than two pollsters on the same day.
* Surveys are conducted for 7 consecutive days per month.
* The occupation of the aircraft is 80 % of the total.
* There is a survey rate of 50 % of the passengers on each selected flight.
* Each survey is done in a limited time of 30 seconds and must end five minutes before the departure of the flight.
* The Canary Islands airports where the surveys are carried out are TFS, TFN, LPA, ACE, FUE, SPC.

Our job is to use mathematical programming and show that it can be quite useful to solve this type of problem.

2. Outline of the first Chapter

The problem proposed by ISTAC is explained to us and we can observe the mathematical model that we use to solve this problem with its restrictions.
Parameters:

- V≡ Number of flight of flights leaving from a Canarian airport.
- K≡ Number of interviewers in a Canarian airport.
- Num_min_V≡ Minimum number of surveys
- Num_Country≡ Number of countries to which the flights are directed.
- J≡ Workday = 8 hours.
- T≡ Minutes it takes to conduct a survey.
- Num_Fli_INE, FlightINE = 1, ..., Nm_Vue_INE, country_INE$_t$, t ∈ VueloINE, $num\_seats\_INE_t$, $hour\_INE_t$, day_INE$_t$, country_INE$_t$, day_INE$_t$, $hour\_INE_t$, $num\_seats\_INE_t$, t ∈ FlightINE.
- Occupation
- Interviews
- Inter = 1, ..., K.
- Flight = 1, ..., V.
- Country = 1, ..., Num_País.
- country$_i$, num_seats_i, $hour_i$, $surveys_i$, day$_i$, $consumption_i$, i ∈ Flight.
- $surveys\_INE_j$, $surveys\_INE_j$, j ∈ Country.

Variables:

- $x_{i,k} = 1, if k should work on the flight i and 0 in other case$, ∀ i ∈ Flight and k ∈ Inter.
- $y_i = 1 if the flight i is interviewed and 0 in other case$, ∀ i ∈ Flight.

The mathematical model is the following:

max $\sum_i surveys_i * y_i$ para todo i∈Flight.
subject to:

1. $y_i \leq L_i, \forall i \in Flight.$
2. $\sum_k x_{i,k} \leq 2 * y_i, \forall i \in Flight, k \in Inter.$
3. $\sum_k x_{i,k} \geq y_i, \forall i \in Flight, k \in Inter.$
4. $For i, j \in$ Flight where $i \neq j$, day$_i$ = day$_j$ and hour$_j \geq hora_i$:
   (a) If $hour_j - hour_i < \frac{consumption_i}{2}, then$:

   $$x_{i,k} + x_{j,k} \leq 1,$$

   ∀ k ∈ Inter.
   (b) If $hour_j - hour_i > J * 60 - \frac{consumption_i}{2}$, $then$:

   $$x_{i,k} + x_{j,k} \leq 1,$$

   ∀ k ∈ Inter.
   (c) If $hour_j - hour_i < consumption_j, then$:

   $$x_{i,k} + x_{j,k} \leq 1 + \sum_l x_{j,l},$$

   ∀ k, l ∈ $Interyl \neq k$.
   (d) If $hour_j - hour_i > j * 60 - consumption_i$, $then$:

   $$x_{i,k} + x_{j,k} \leq 1 + \sum_l x_{i,l},$$

   ∀ k, l ∈ $Interyl \neq k$.
5. $\sum_i surveys_i * y_i \geq Num\_Min\_V_j, \forall i \in$ Flight, j ∈ Country y j = country$_i$.
6. $x_i, k \in 0, 1, \forall i \in Flighty \forall k \in Inter.$
7. $y_i \in 0, 1, \forall i \in Flight.$

3. Outline of the second chapter

In this section we are going to talk about several types of linear programming problems, which are related to our particular problem.

4. Outline of the third chapter

We implemented a C code to create a file that shows the flights of a particular Canary Island airport for a whole week, and then, using the algorithm in CPLEX, we obtain the optimal solution to the problem.

References

[1] J.J. Salazar. *Programación Matemática. Editorial Díaz de Santos, 2001.*

[2] Archivo de datos Aeropuertos Canarios. *Aeropuertos_Canarios.xls.*

[3] ISTAC. Estadística de movimientos turísticos en fronteras de Canarias (FRONTUR-Canarias). *http://www.gobiernodecanarias.org/istac/galerias/documentos/E16028B/metodologia_FRONTUR.pdf.*

[4] Matteo Fischetti, Silvano Martello y Paolo Toth. *The fixed job schedule problem with spread-time constraints.*

[5] Ilya Gertsbakh y Helman I. Stern. *Minimal resources for fixed and variable job schedules.*

[6] Guía de aviones comerciales. *https://www.emptyleg.com/es/guia-de-aviones/guia-aviones-comerciales.*