Universidad de La Laguna

Sección de Matemáticas | Facultad de Ciencias

Serezade Tedauit González Torres

Un problema de rutas que le surge a una compañía de aguas de abastecimiento público

Problemas de Rutas

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

Dirigido por

Hipólito Hernández Pérez

Hipólito Hernández Pérez
Departamento de Estadística e Investigación Operativa
Universidad de La Laguna
38271 La Laguna, Tenerife

Agradecimientos

A Hipólito Hernández Pérez,
por animarme a llevar a cabo este proyecto,
y por su dedicación y ayuda para elaborar esta memoria.
A mi familia,
por su apoyo y amor incondicional.
A mis amigos,
por acompañarme durante estos años.

Resumen · Abstract

Resumen

Los problemas de rutas son una de las ramas más destacadas de la Investigación Operativa. Se trata de problemas que surgen de forma natural en una gran variedad y cantidad de ámbitos. Teniendo en cuenta esto, la siguiente memoria expone un problema de rutas que nace en una empresa pública de abastecimiento de aguas. Se quieren optimizar las rutas realizadas por los trabajadores encargados de la lectura de contadores. Para llevar a cabo el planteamiento de este problema se mostrará de forma breve un estudio de los principales problemas de rutas: el problema del viajante de comercio, el problema del cartero chino y el problema de enrutamiento de vehículos. Esto nos servirá para facilitar formulación del modelo llevado a cabo. Dicho modelo ha ido madurando a lo largo del análisis del problema hasta obtener uno que reflejase lo más posible el problema real, y lo resolviese de manera viable computacionalmente hablando.

Palabras clave: *Problemas de rutas – Investigación Operativa – Problema del cartero chino – Problema de enrutamiento de vehículos*

Abstract

Routing problems are one of the most outstanding branches of Operational Research. These problems arise naturally in a large variety and number of areas. Taking this into account, the following report exposes a routing problem that is born in a public water supply company, aiming to optimize the routes made by the workers in charge of reading meters. To carry out the approach of this problem, a study of the main problems of routes will be shown briefly: travelling salesman problem (TSP), Chinese postman problem and vehicle routing problem (VRP). This will facilitate the formulation of the model carried out. This model has matured developed until obtaining a model that reflects the real problem as much as possible and solves it in a computationally viable way.

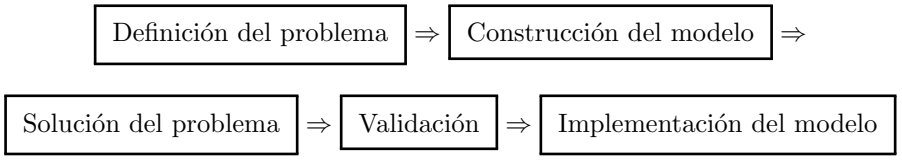
Keywords: *Routing problems – Operational Research – Chinese postman problem – Vehicle routing problem*

Contenido

Introducción

La Investigación Operativa es una ciencia moderna que utiliza modelos matemáticos, estadísticos y algoritmos para formular y resolver problemas complejos, obteniendo así soluciones óptimas de los problemas y facilitando la toma de decisiones. Actualmente la Investigación Operativa incluye gran cantidad de ramas como la Programación Lineal, Programación No Lineal, Programación Dinámica, Simulación, Teoría de Colas, Teoría de Inventarios, Teoría de Grafos, etc. El nacimiento como ciencia de esta rama de las matemáticas se establece durante la Segunda Guerra Mundial y debe su nombre a las operaciones militares. Sin embargo, se podría considerar que los verdaderos orígenes de la Investigación Operativa se remontan mucho más atrás en el tiempo, ya que el problema de hacer un uso óptimo de los recursos disponibles ha existido siempre. Es importante destacar que el avance de esta ciencia va ligado al avance informático, ya que este ha posibilitado la resolución de problemas en la práctica y la obtención de soluciones que de otra forma conllevarían un enorme tiempo de cálculo haciéndolos inviables. Debido al gran éxito obtenido en el campo militar, esta rama de las matemáticas se extendió a otros campos tales como la industria, física, administración, informática, ingeniería, economía, estadística y probabilidad, ecología, educación, servicio social, ..., siendo hoy en día utilizada prácticamente en todas las áreas imaginables donde se pretenda mejorar la eficiencia.

En general, el enfoque de la Investigación Operativa es el modelaje. Crear un modelo que será usado como una herramienta analítica que nos sirva para lograr una visión bien estructurada de la realidad. Así, el propósito del modelo es proporcionar un medio para analizar el comportamiento de las componentes de un sistema con el fin de optimizar su desempeño. Una visión esquemática del proceso asociado a la construcción de un modelo de optimización es:



Motivación y objetivos

Tal y como se ha comentado anteriormente, la Investigación Operativa se utiliza, entre otras cosas, para la optimización de rutas en diversos casos, como por ejemplo repartir el correo, las rutas de los aviones, rutas de camiones con cargas, etc. A lo largo de este trabajo trataremos de llevar a cabo también una optimización de rutas desarrollando un modelo matemático con el que obtener la mejor ruta que debería de realizar un lector de contadores de una empresa pública de abastecimiento de agua (mostraremos un ejemplo contextualizado). La idea de resolver este problema surgió a lo largo de las prácticas externas realizadas en la Empresa Mixta de Aguas de Santa Cruz de Tenerife, donde se planteó la posibilidad de mejorar las rutas que los lectores realizaban.

1

Problemas de rutas conocidos

Tal y como se ha comentado en la introducción, nuestro problema se trata de un problema de rutas, una de las ramas más importantes de la Investigación Operativa. Por ello antes de comenzar a trabajar en nuestro modelo veremos algunos de los problemas de rutas más destacados y que tienen relación con el nuestro. Comenzaremos con el problema del viajante de comercio, conocido en inglés como "Traveling Salesman Problem"(TSP) debido a que es muy importante y uno de los problemas más estudiados de la optimización, seguidamente hablaremos del problema del cartero chino debido a la similitud que su planteamiento puede llegar a tener con el del problema a modelar, y por último trataremos el problema de enrutamiento de vehículos (VRP, por su siglas en inglés) que ha sido de gran utilidad para realizar uno de los modelos que hemos formulado a lo largo del trabajo.

1.1. Problema del viajante del comercio

El problema del viajante o TSP consiste en conseguir una ruta óptima para recorrer un conjunto de ciudades sin repetir ninguna de ellas y volviendo al lugar de origen. Este problema es uno de los más importantes y estudiados de la optimización combinatoria. A pesar de la sencillez de su enunciado se trata de un problema bastante complicado, y aunque sabemos que tiene solución, la dificultad de encontrar la solución óptima en este problema reside en que pertenece a la clase de problemas NP-duros, que son aquellos que no tienen algoritmos para resolverlos en tiempo polinomial.

Uno de los motivos por el que el TSP es muy estudiado es porque está presente y se puede extrapolar a muchos otros problemas más generales que se encuentran en la práctica de forma habitual, como por ejemplo los problemas de enrutamiento.

Una definición bastante genérica y clara del TSP la podemos encontrar en el libro de Davendra [1], es la siguiente:

Definición 1.1. *Dado un conjunto de ciudades y el coste (o distancia) de viajar entre cada par posible de estas ciudades, el problema del viajante de comercio consiste en encontrar el mejor camino posible que visite todas las ciudades una única vez y vuelva al punto de partida minimizando el coste del viaje.*

Otros referencias centradas en el estudio del TSP son Flood [2] y Gutin y Punnen [3].

Matemáticamente una de las formulaciones del TSP haciendo uso de la programación lineal entera es la siguiente. Tomamos $V = \{0, 1, \ldots, n\}$ el conjunto de n ciudades y a c_{ij}, con $i, j \in V$ al costo de ir de una ciudad a otra. Por otro lado definimos las variables

$$x_{ij} = \begin{cases} 1 & \text{si existe el camino de } i \text{ a } j, \\ 0 & \text{en otro caso,} \end{cases} \qquad \forall i, j \in V$$

$$u_i = \text{ representan el lugar de la secuencia en el que se visita el nodo } i \qquad \forall i \in V$$

y el modelo correspondiente sería

$$\text{mín} \sum_{i=0}^{n} \sum_{\substack{i=0 \\ i \neq j}}^{n} c_{ij} \cdot x_{ij} \tag{1.1}$$

sujeto a:

$$\sum_{i=0}^{n} x_{ij} = 1 \qquad \forall j \in V \tag{1.2}$$

$$\sum_{j=0}^{n} x_{ij} = 1 \qquad \forall i \in V \tag{1.3}$$

$$u_i - u_j + n x_{ij} \leq n - 1 \qquad \forall i, j \in V, i \neq 0 \tag{1.4}$$

1.2. El problema del cartero chino

El problema del cartero chino o problema del circuito del cartero es el primer problema de rutas por arcos en el que se plantea la posibilidad de construir un ciclo euleriano (un camino cerrado que alcance toda arista una vez) con coste óptimo. Fue originalmente estudiado por el matemático chino Mei-Ko Kuan en 1962, quien precisamente era cartero y lo que él planteaba era el problema al que se enfrenta el cartero para repartir la correspondencia recorriendo la menor

distancia posible. Matemáticamente este problema consiste en encontrar el camino óptimo, es decir, el camino más corto con el que se visite cada arista de un grafo al menos una vez volviendo al punto (o nodo) de partida. El problema original dio lugar a multitud de variantes, algunas de ellas son:

- Problema del cartero chino para grafos dirigidos: se recorren arcos en los que únicamente se permite seguir una dirección determinada.
- Problema del cartero chino para grafos no dirigidos: se recorren aristas y puede hacerse en cualquier dirección.
- Problema del cartero chino para grafos mixtos: en esta se mezclan aristas, que pueden ser recorridas en cualquier dirección, y arcos, que han de ser recorridos en la dirección permitida.
- Problema del cartero $k-$Chino: encontrar todos los ciclos de k elementos a partir de un local designado, de tal forma que cada arista sea atravesada por lo menos por un ciclo. El objetivo es minimizar el costo del ciclo más caro.
- Problema del cartero rural: dado un subconjunto de aristas, encontrar el ciclo hamiltoniano* más barato conteniendo una de esas aristas (y posiblemente otras).

Respecto a las variantes nombradas, nótese que no se conocen algoritmos polinomiales que las resuelvan de forma exacta, denominándose por lo tanto problemas del tipo NP-duros.

El libro de Corberán y Laport [4] estudia este problema y otras variantes, y en él podemos encontrar información más detallada al respecto.

1.3. Problema de enrutamiento de vehículos

El problema de enrutamiento de vehículos o VRP es un problema de Optimización Combinatoria que consiste en calcular el conjunto óptimo de rutas para una flota de vehículos que debe satisfacer las demandas de sus clientes. Este se trata de una generalización del TSP. El problema requiere la entrega de cierto producto, almacenado en un único local, a los clientes que realizan cierta demanda; el objetivo fundamental es minimizar el coste total de las rutas trazadas. Se trata de un problema NP-duro, por ello las implementaciones más utilizadas para resolver el problema se basan en heurísticas debido a que para grandes instancias del problema, como sucede en ejemplos reales, producen buenos resultados. Podemos encontrar una descripción más exhaustiva de este problema en el libro de P. Toth y D. Vigo [5].

Dado el problema que se quiere resolver cabe destacar que la función objetivo del VRP puede ser muy diferente según su aplicación, se puede querer

* Un ciclo hamiltoniano es una sucesión de aristas adyacentes, que visita todos los vértices del grafo una sola vez y además el último vértice visitado es adyacente al primero.

minimizar la distancia total de las rutas, el número de vehículos utilizados, la variación entre el tiempo del viaje, la carga del vehículo, etc. Y por tanto, como se puede deducir, existen muchas variantes de este problema, entre las que podemos destacar:

- Problema de enrutamiento del vehículo con recogida y entrega (VRPPD): necesitamos trasladar productos desde una ciertas ubicaciones de recogida hasta los correspondientes puntos de entrega. El objetivo es encontrar las rutas óptimas para una flota de vehículos que realicen dicha tarea.
- Problema de enrutamiento del vehículo con ventanas de tiempo (VRPTW): las ubicaciones de entrega tienen ciertos periodos de tiempo o ventanas de tiempo durante los cuales las entregas deben de realizarse.
- Problema de enrutamiento del vehículo con capacidad (CVRP): los vehículos tienen un límite de capacidad para transportar los productos que deben ser entregados.
- Problema de enrutamiento del vehículo con viajes múltiples (VRPMT): los vehículos pueden hacer más de una ruta.
- Problema de enrutamiento de vehículo abierto (OVRP): las rutas de los vehículos no tienen porqué terminar en el depósito.

2

Formulación del problema

Para llevar a cabo un modelo que solventase nuestro problema fue necesario un buen planteamiento del mismo. Este nos llevó entre otras cosas a plantear diversas variantes. Será esto, el planteamiento y las variantes del modelo, lo que se exponga a lo largo de este capítulo.

2.1. Primer planteamiento

El problema surge de la necesidad de planificar rutas para los trabajadores, de una empresa pública de abastecimiento de aguas, que leen los contadores de las casas de la capital tinerfeña. A grandes rasgos, se tratará de buscar las mejores rutas, que son aquellas que nos permitan leer todos los contadores en el menor tiempo posible. Así, el objetivo es minimizar el tiempo en el que se realizan las lecturas. Cabe destacar que esto nos lleva a plantearnos diversas cuestiones en cuanto al planteamiento del problema, que se convertirán en variantes del mismo. En primer lugar, cabe la posibilidad de que se quiera realizar el trabajo por una única persona para lo que se creó la primera variante 2.2.1. Seguidamente se planteó la posibilidad de que no fuese una, sino varias, las personas que llevasen a cabo el trabajo, por lo que se creó la segunda 2.2.2.

Dado el problema general planteado tenemos los siguientes parámetros comunes en cada una de las opciones:

- Conjunto de n esquinas de la zona a estudiar $\equiv V = \{0, 1, 2, 3, ..., n\}$.
- Siendo $0 \in V$ el depósito.
- El costo de ir de la esquina i a la j (de recorrer una calle) $\equiv c_{i,j}$, con $i, j \in V$.
- El número de casas en una calle $\equiv h_{ij}$, con $i, j \in V$.
- Definimos

$$l_{ij} = \begin{cases} 1 & \text{si } [i,j] \text{ hay que recorrerlo obligatoriamente.} \\ 0 & \text{en otro caso} \end{cases}$$

- C_1 es una constante que corresponde al tiempo que tarda un operario en descargar la información de un contador.

Para los distintos modelos presentados tomaremos las variables:

$$\begin{aligned} x_{ij} &= \begin{cases} 1 & \text{si se va en la ruta de } i \text{ a } j, \\ 0 & \text{en otro caso,} \end{cases} && \forall i, j \in V \\ y_i &= \text{número de veces que se visita el nodo } i && \forall i \in V, \\ f_{ij} &= \text{flujo que va de } i \text{ a } j && \forall i, j \in V. \end{aligned}$$

Nótese que estas variables sufrirán ligeros cambios en algunas de las variantes que se presentarán. La familia de variables f_{ij} la utilizaremos para evitar subciclos.

2.2. Variantes

2.2.1. Resolución del problema por una persona

Expondremos a continuación el modelo en el que solo tenemos en cuenta una persona, es decir, un único lector de contadores sale de la empresa, el depósito, y recorre las calles obligatorias, aquellas en las que debe realizar lectura, en la menor cantidad de tiempo posible.

$$\text{mín} \sum_{\substack{i,j \in V \\ i \neq j}} c_{ij} \cdot x_{ij} + \sum_{\substack{i,j \in V \\ i \neq j}} C_1 \cdot h_{ij} \cdot l_{ij} \tag{2.1}$$

sujeto a:

$$\sum_{\substack{j \in V \\ j \neq i}} x_{ij} = y_i \qquad \forall i \in V \tag{2.2}$$

$$\sum_{\substack{j \in V \\ j \neq i}} x_{ji} = y_i \qquad \forall i \in V \tag{2.3}$$

$$x_{ij} + x_{ji} \geq l_{ij} \qquad \forall i, j \in V, i \neq j \tag{2.4}$$

$$y_0 = 1 \tag{2.5}$$

$$\sum_{\substack{j \in V \\ j \neq i}} f_{ij} - \sum_{\substack{j \in V \\ j \neq i}} f_{ji} = y_i \qquad \forall i \in V, i \neq 0 \tag{2.6}$$

$$f_{ij} \leq 2 \cdot n \cdot x_{ij} \qquad \forall i, j \in V, i \neq j \tag{2.7}$$

$$f_{ij} \geq 0 \qquad \forall i, j \in V, i \neq j \tag{2.8}$$

Tal y como se puede observar en este modelo disponemos de una función objetivo (2.1) que trata de minimizar el coste del problema planteado. En este caso se tratará el coste como el total de metros recorridos. Contemplándose en el mismo tanto el coste de caminar cada una de las calles necesarias para recorrer todas aquellas que definimos con l_{ij} como obligatorias y, además, el coste que supone detenerse en cada una de las casas a realizar la lectura (la segunda parte de la función objetivo). El tiempo que se tarde en cada casa se ha traducido en metros para mantener la misma unidad, se ha tenido en cuenta, según la velocidad promedio a la que camina un ser humano, que 1 minuto equivale a 83 metros caminados. Así la constante C_1 se ha definido a 415 (que equivale a 5 minutos).

Las restricciones (2.2) y (2.3) fuerzan a que el número de arcos que salen y que entran (respectivamente) del nodo i coinciden con y_i. Las restricciones (2.4) fuerzan que se pase por los arcos que tienen contadores que visualizar indistintamente del sentido que se elija. Es decir, cada lado de la calle se considera como arcos por separado, de modo que en cada una hay cuatro arcos, dos correspondientes a un lado y dos a otro, al igual que se considera esquina *nodo* a cada extremo de cada uno de los lados de la calle. Con la restricción (2.5) obligamos a que se comience en la empresa y se termine en esta, intentando de este modo simular lo más posible la realidad. Por último, las restricciones (2.6)–(2.8) hacen uso de las variables de flujo para evitar subciclos.

En el siguiente grafo, Figura 2.1 podemos observar cómo se representaría una calle:

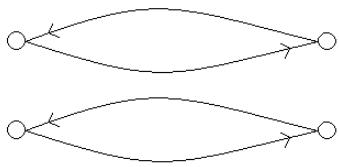


Figura 2.1: Grafo del recorrido de una calle

2.2.2. Resolución del problema por k personas

Desarrollaremos a continuación un modelo en el que tenemos la misma finalidad pero en este caso varias personas realizaran los recorridos. Para realizar este, cabe destacar que las variables cambiarían ligeramente y se definirían como:

$$x_{ij}^k = \begin{cases} 1 & \text{si la persona } k\text{-ésima va de } i \text{ a } j. \\ 0 & \text{en otro caso} \end{cases} \qquad \forall i, j \in V, k \in P,$$

$y_i^k =$ número de veces que la persona k visita el nodo i $\quad \forall i \in V, k \in P,$

$f_{ij}^k =$ flujo de la persona k que va de i a j $\quad \forall i, j \in V k \in P.$

Donde definimos un nuevo conjunto $P = \{1, \ldots, p\}$ que corresponde al número de personas. Además añadiremos una nueva variable, esta nos servirá para distinguir cuales son las calles en las que cada persona realiza la lectura de los contadores, ya que, tal y como se plantea el problema es posible que distintas personas pasen por una misma calle. Dicha variable es:

$$z_{ij}^k = \begin{cases} 1 & \text{si la persona } k\text{-ésima lee de } i \text{ a } j \\ 0 & \text{en otro caso.} \end{cases}$$

Teniendo en cuenta lo anterior el modelo es el siguiente:

$$\text{mín} \sum_{\substack{i,j \in V \\ i \neq j \\ k \in P}} x_{ij}^k \cdot c_{ij} + \sum_{\substack{i,j \in V \\ i \neq j \\ k \in P}} C_1 \cdot h_{ij} \cdot l_{ij} \tag{2.9}$$

sujeto a:

$$\sum_{\substack{j \in V \\ j \neq i}} x_{ij}^k = y_i^k \quad \forall i \in V, \forall k \in P \tag{2.10}$$

$$\sum_{\substack{j \in V \\ j \neq i}} x_{ji}^k = y_i^k \quad \forall i \in V, \forall k \in P \tag{2.11}$$

$$\sum_{k \in P} (z_{ij}^k + z_{ji}^k) = l_{ij} \quad \forall i, j \in V, i \neq j \tag{2.12}$$

$$y_0^k = 1 \quad \forall k \in P \tag{2.13}$$

$$\sum_{\substack{j \in V \\ j \neq i}} f_{ij}^k - \sum_{\substack{j \in V \\ j \neq i}} f_{ji}^k = y_i^k \quad \forall i \in V, i \neq 0, \forall k \in P \tag{2.14}$$

$$f_{ij}^k \leq \left(\sum_{\substack{i,j \in V \\ i < j}} l_{ij} \right) \cdot \frac{3}{p} \cdot x_{ij}^k \quad \forall i, j \in V, i \neq j, \forall k \in P \tag{2.15}$$

$$f_{ij}^k \geq 0 \quad \forall i, j \in V, i \neq j, \forall k \in P \tag{2.16}$$

$$x_{ij}^k \geq z_{ij}^k \quad \forall i, j \in V, i \neq j, \forall k \in P \tag{2.17}$$

$$c(k) = \sum_{\substack{i,j \in V \\ i \neq j}} c_{ij} \cdot x_{ij}^k + \sum_{\substack{i,j \in V \\ i \neq j}} C_1 \cdot z_{ij}^k \quad \forall k \in P \tag{2.18}$$

$$c(k_1) - c(k_2) \leq C_2 \quad \forall k_1, k_2 \in P, k_1 \neq k_2 \tag{2.19}$$

En este modelo se trata de optimizar del mismo modo que en el explicado anteriormente, a pesar de ser más personas las que realizan el trabajo. En las cuatro primeras restricciones tratamos de recorrer las calles necesarias para cubrir todas las obligatorias, de forma que cada persona que salga de la empresa vuelva a ella habiendo cumplido dicha tarea. Por otro lado, tenemos las restricciones (2.14)–(2.16) con las que evitamos que se formen subciclos. Y por último, queriendo que el modelo se acerque lo más posible a la realidad, se han creado las tres últimas familias de restricciones con las que se trata de compensar el trabajo de cada lector (si así se requiriera), es decir, que el recorrido hecho por cualesquiera dos de ellos no tenga un coste que difiera más de la constante C_2.

Para aplicar los modelos a la empresa tendríamos que utilizar el problema de k personas, ya que en nuestro caso se dispone de un número de trabajadores que, durante su jornada laboral, es decir, un tiempo determinado, deben llevar a cabo las lecturas de todas las casas. Sin embargo, tal y como observaremos más adelante en las tablas de los resultados computacionales, este modelo no es viable para una alta cantidad de nodos ni de personas. Por ello plantearemos una alternativa a dicho modelo en la siguiente sección.

2.3. Segundo planteamiento

2.3.1. Resolución del problema por zonas

En este caso vamos a plantear resolver el problema en dos fases, de modo que, en primer lugar dividiremos el territorio a analizar en zonas no muy grandes, aplicando en cada una de ellas el modelo creado para realizar la lectura por una sola persona 2.2.1. A continuación, crearemos un nuevo modelo en el que k personas recorran dichas zonas, todo ello en la menor cantidad de tiempo posible. Cabe destacar que realizar las rutas en el menor tiempo posible es equivalente a minimizar las distancias recorridas por cada uno de los empleados o lectores. Pues bien, para desarrollar el modelo de zonas usaremos los siguientes conjuntos y parámetros:

- Conjunto de n zonas a visitar $\equiv V = \{0, 1, 2, 3, ..., n\}$.
- Siendo $0 \in V$ el depósito.
- El costo de ir de la zona i a la $j \equiv c_{ij}$, con $i, j \in V$.
- La distancia que se recorre en una zona $i \equiv d_i$, con $i \in V$.
- La cantidad de metros que una persona puede recorrer en su jornada laboral de 8 horas equivale a Q =35000 metros.

Tomamos las variables

$$x_{ij} = \begin{cases} 1 & \text{si vamos de i a j.} \\ 0 & \text{en otro caso} \end{cases}$$
$$f_{ij} \geq 0 \quad \text{variable flujo.}$$

y el modelo correspondiente es:

$$\text{mín} \sum_{\substack{i,j \in V \\ i \neq j}} c_{ij} \cdot x_{ij} + \sum_{i \in V} d_i \tag{2.20}$$

sujeto a:

$$\sum_{\substack{j \in V \\ i \neq j}} x_{ij} = 1 \qquad \forall i \in V, i \neq 0 \tag{2.21}$$

$$\sum_{\substack{j \in V \\ i \neq j}} x_{ji} = 1 \qquad \forall i \in V, i \neq 0 \tag{2.22}$$

$$\sum_{\substack{i \in V \\ i \neq 0}} x_{i0} = \sum_{\substack{j \in V \\ j \neq 0}} x_{0j} \tag{2.23}$$

$$\sum_{\substack{j \in V \\ j \neq i}} f_{ij} - \sum_{\substack{j \in V \\ j \neq i}} f_{ji} = d_i \qquad \forall i \in V, i \neq 0 \tag{2.24}$$

$$f_{ij} \leq Q \cdot x_{ij} \qquad \forall i, j \in V, i \neq j \tag{2.25}$$

$$f_{ij} \geq 0 \qquad \forall i, j \in V, i \neq j \tag{2.26}$$

Con este modelo se trata de minimizar el costo de visitar todas las zonas, teniendo como costo final el costo del recorrido de ir de una zona a otra más el peso de cada una de las zonas (d_i), que es la cantidad de metros que se recorren en dicha zona que es el resultado de aplicar el problema de una persona, tal y como se ha dicho con anterioridad. Por otra parte, las restricciones del modelo tratan de hacer que se recorran todas las zonas partiendo de la empresa y llegando a la misma. Cabe destacar la restricción (2.23) la cual nos indica que se puede salir tantas veces como sea necesario de la empresa pero, se ha de entrar al final el mismo número de veces. Dicho número de veces corresponde al número de personas necesarias para recorrer todas las zonas durante la jornada laboral (Q). Finalmente, al igual que en los modelos anteriores, las restricciones que contienen la variable flujo las usamos para evitar subciclos, aunque en este caso también la restricción (2.25) nos sirve para que cada trabajador vuelva de nuevo a la empresa sin sobrepasar Q, su jornada de trabajo.

Nótese que este último modelo corresponde a una de las variantes del VRP, nombradas en el capítulo 1, que se trata de una formulación del enrutamiento de vehículos con capacidades. Se debe mencionar que en este no se tiene en

cuenta el costo entre zonas para que el trabajador vuelva a la empresa antes del final de su horario. Esto se debe a que previamente se utilizó la variante VRP *Distance-Constraint* en la que sí se contemplaba este detalle, pero la tardanza computacional del mismo nos llevó a optar por la opción expuesta. Esta diferencia la hemos compensando en el modelo disminuyendo levemente la constante Q, ya que el costo de ir de una zona a otra es bastante pequeño en comparación con el peso de cada zona.

Todos estos modelos han sido programados y resueltos para ejemplos aleatorios, además el segundo planteamiento del problema se ha aplicado también a un ejemplo real en una zona de S/C de Tenerife tal y como observaremos en el próximo capítulo.

3

Resultados computacionales y problema real

3.1. Programas informáticos

Hay muchos programas que resuelven problemas de programación lineal entera entre ellos GUSEK y XPRESS. Por una parte, GUSEK se trata de un *sofware* libre basado en el uso de paquetes de modelización como el GLPK (GNU Linear Programming Kit) y pensado para la programación a gran escala de problemas lineales, enteros-mixtos y de otros problemas relacionados. Por otra parte, XPRESS no es un *sofware* libre pero sí más potente, se trata de entorno de programación matemática confeccionado para aportar capacidades de resolución de alto rendimiento.

Al comienzo de la programación de nuestros modelos se usó GUSEK, porque el hecho de ser un programa gratuito resulta de interés para resolver los problemas. Sin embargo, su potencia no era suficiente para llevar a cabo la tarea requerida. Por ello se consideró oportuno el uso de un programa con más potencia, el XPRESS. Con este se realizó la resolución del problema planteado de dos maneras, de forma aleatoria y aplicado a una simulación real, tal y como expondremos en las siguientes secciones.

3.2. Resultados para una persona

Tras programar el modelo de una persona en XPRESS produciendo los datos de forma aleatoria se han estudiado casos distintos, generando los datos con diversas semillas y con distintas cantidades de nodos. Los datos obtenidos, serán reflejados en la Tabla 3.1.

En esta se muestra para 15, 20 y 25 nodos los costos y tiempo de resolución con distintas semillas. Además también se contabilizan en cada uno de los casos el total de calles por las que hay que pasar obligatoriamente.

Resultados aleatorios				
Nodos	Semilla	Número de l_{ij}	Costo	Tiempo
15	1	15	17124	0.8
	2	12	15757	0.7
	3	13	13792	0.1
20	1	22	26390	1.1
	2	18	22990	0.8
	3	27	32671	0.2
25	1	30	37324	0.9
	2	28	34397	1.2
	3	32	39018	1.5

Tabla 3.1: Resultados de casos aleatorios para una persona.

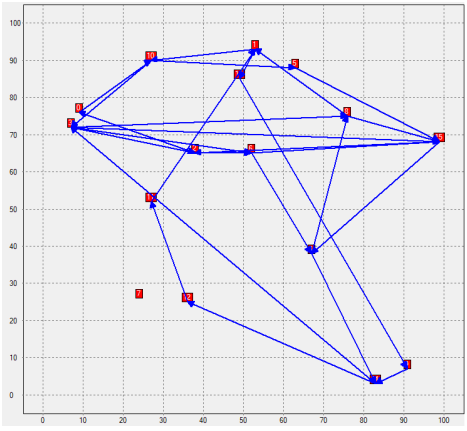


Figura 3.1: Resultados un ejemplo aleatorio para una persona.

En la siguiente Figura 3.1 se muestra uno de los casos de la tabla.

En cuanto al gráfico de la figura, destaca el hecho de encontrar un nodo no conectado, esto se debe a que si ninguna de las calles obligatorias llegaba a ese nodo no es necesario ir a él. También debemos mencionar que las coordenadas de cada nodo se han generado de forma aleatoria y los costos de ir de uno a otro se han establecido con el uso de la distancia *Manhattan*.

3.3. Resultados para k personas

Al igual que la anterior variante, el modelo de k personas ha sido programado en XPRESS. Mostraremos a continuación en la Tabla 3.2 algunos de los

resultados obtenidos al llevarlo a cabo de forma aleatoria, con distintas semillas y distintos números de personas.

Ejemplos generados aleatoriamente				
N° de personas	Nodos	Semilla	Costo	Tiempo
2	10	1	21240	2.4
		2	18335	1.2
		3	15584	919.5
	12	1	25474	0.7
		2	21284	1.4
		3	19267	2.7
	14	1	24331	6.5
		2	24359	5863.2
		3	20544	45.2
3	8	1	18407	22.7
		2	17523	1.4
		3	19684	21.9
	10	1	21320	627.1
		2	18457	2414.1
		3	26775	2.2
	11	1	22599	966.6
		2	18078	129.2
		3	27214	7.8

Tabla 3.2: Resultados de casos aleatorios para k personas.

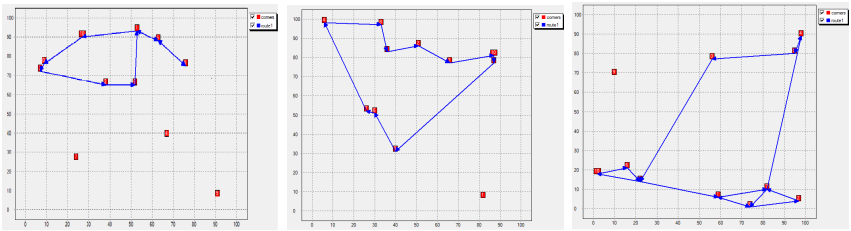
Aquí se muestra para dos y tres personas cuál es el costo y el tiempo de resolución según el número de nodos y para distintas semillas.

3.4. Resultados del problema por zonas

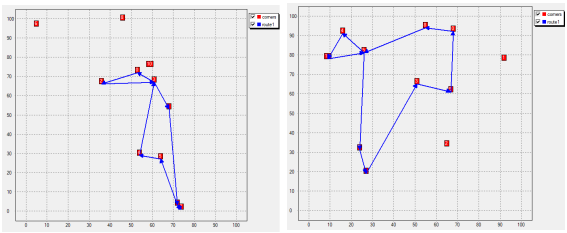
Dado este planteamiento vamos a generar un problema aleatoriamente usando las dos fases indicadas. Tomaremos 5 zonas aleatorias de 10 nodos cada una (las obtendremos con semillas diferentes), o lo que es lo mismo 10 esquinas. La Figura 3.2 muestra las soluciones de cada una de estas 5 zonas. Seguidamente con los datos obtenidos aplicaremos el modelo de zonas, que leerá los datos de una persona de cada zona y generará las distancias entre zonas de forma aleatoria. A continuación, veremos los recorridos de cada zona, así como la ruta final de cada persona.

Ruta de cada una de las 5 zonas

Los datos de resolución de cada una de las zonas son los mostrados en la Tabla 3.3 (cada zona tiene 10 nodos):



(a) ZONA1 (b) ZONA2 (c) ZONA3



(d) ZONA4 (e) ZONA5

Figura 3.2: Gráfica de la primera fase del problema de zonas.

Resultados aleatorios			
ZONA	Semilla	Tiempo de resolución	Costo
1	1	0.0	8115
2	2	0.1	8610
3	3	0.3	11681
4	4	0.2	5214
5	4	0.2	7776

Tabla 3.3: Resultados de la primera fase del problema de zonas.

Según los costos de cada zona y generando de forma aleatoria las distancia entre ellas, obtenemos que la ruta final entre las mismas es la mostrada en la Figura 3.3.

Se observa que salen dos personas del depósito y sus rutas son:

- **Persona 1**: $0 \rightarrow 1 \rightarrow 5 \rightarrow 4 \rightarrow 2 \rightarrow 0$
- **Persona 2**: $0 \rightarrow 3 \rightarrow 0$

Siendo 0 la empresa, el costo de la persona uno es de 29901 metros y el de la dos es de 11693 metros.

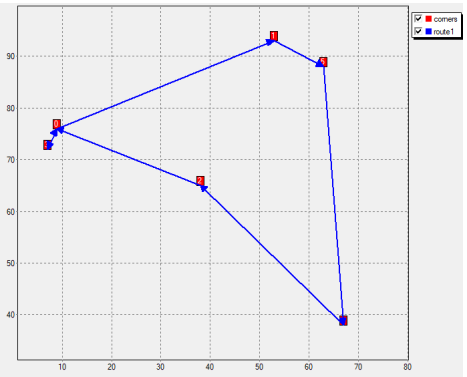


Figura 3.3: Gráfica de la segunda fase del problema de zonas.

3.5. Problema real

A lo largo de esta sección aplicaremos los modelos desarrollados en el Capítulo 2, concretamente el modelo por zonas, en un problema real. Hemos tomado una parte de S/C de Tenerife y hemos llevado a cabo la aplicación del modelo en la misma.

En la Figura 3.4 se muestra un mapa con el callejero del área utilizada, la cual está rodeada por una línea azul. Dicho mapa se ha obtenido con el visor de GRAFCAN*, que también hemos utilizado para recoger los datos de este problema. Con él hemos medido las distancias y calculado el número de casas de cada calle (por lo que estos datos pueden no ser cien por cien exactos). A partir del gráfico dibujado en el mapa se han numerado cada una de las esquinas, obteniéndose un total de 182 .

* GRAFCAN es una empresa pública que realiza actividades de producción, mantenimiento y gestión de la información geográfica y territorial de la Comunidad Autónoma de Canarias.

Figura 3.4: Mapa real de una zona de S/C de Tenerife.

Una vez recopilado todo el conjunto de datos necesarios se dividió el mapa en las zonas mostradas en la Figura 3.5 y se aplicó el modelo para una persona en cada una de ellas, obteniendo los resultados expuestos en la Tabla 3.4.

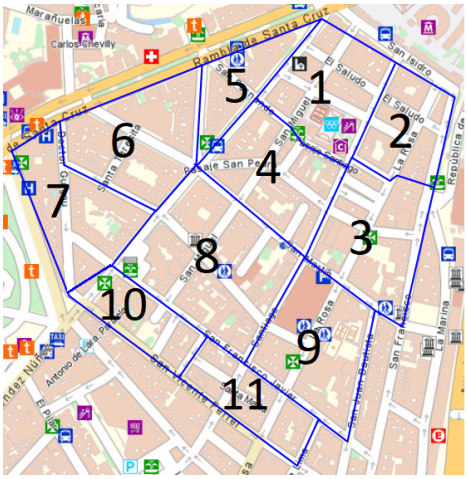


Figura 3.5: Zonas del problema real.

ZONA	Costo
1	11163.1
2	9971,05
3	13431,2
4	10479,7
5	6527,8
6	8797,7
7	10544,2
8	13660,6
9	7465
10	7090,5
11	9375,4

Tabla 3.4: Costos por zona del problema real.

Tras guardar estos datos, las distancias entre cada una de las zonas, las cuales se tomaron de centro a centro, y la distancia de cada zona al depósito (se ha considerado como tal una empresa de abastecimiento de aguas de S/C de Tenerife), se aplica el modelo de zonas con dichos datos y obtenemos que 4 personas deben salir de la empresa a realizar sus correspondientes rutas y que el costo óptimo de todas ellas será de 124527 metros. Cabe destacar que el tiempo computacional para resolver el modelo de zonas ha sido de 2214,7 segundos.

En la siguiente tabla de contenidos (Tabla 3.5) reflejaremos la ruta que debe seguir cada una de las cuatro personas y el costo de cada una de ellas.

Rutas de zonas por persona		
Persona	Ruta	Costo
1	0-1-2-3-0	39002
2	0-7-0	14012
3	0-10-4-5-6-0	37050
4	0-11-9-8-0	34463

Tabla 3.5: Resultados de la segunda fase del problema de zonas.

Por último, en la Figura 3.6 vemos la ruta de zonas seguida por cada trabajador.

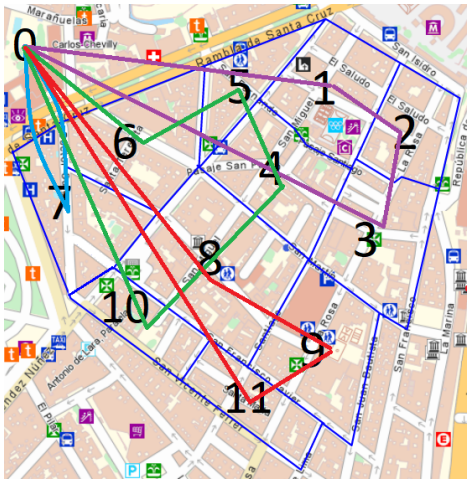


Figura 3.6: Gráfica de la segunda fase del problema de zonas.

Nótese que el depósito no está situado en la figura en su localización real, pero se ha hecho así para una mayor claridad.

4

Conclusiones

En esta memoria hemos estudiado la optimización de rutas para los trabajadores que leen contadores en las empresas de abastecimiento de aguas. Dicho problema surgió a lo largo de la realización de las prácticas externas en la Empresa Mixta de Aguas de S/C de Tenerife. Al tratarse de un planteamiento abierto se han ido realizando modelos que se aproximasen lo más posible a la realidad, estos se han programado inicialmente en GUSEK y más tarde en XPRESS debido a la necesidad de un programa con mayor potencia. Tras ver que lo formulado funcionaba en casos aleatorios se eligió el modelo que computacionalmente era más eficiente. Después se decidió recopilar un conjunto de datos reales y aplicarlo en estos. De ellos se han obtenido unas rutas determinadas para una zona de la capital tinerfeña, y debido a como se ha planteado y modelado el problema podríamos aplicarlo a las zonas necesarias en casos reales de distintas empresas de abastecimiento de aguas.

Sin embargo, una propuesta de mejora sería modificar el modelo 2.3.1 para que sea flexible y dentro de cada zona podamos empezar y terminar en puntos diferentes, e ir de los puntos finales de cada zona a los iniciales de las otras. También sería de interés haber aplicado el modelo en datos reales de una empresa de aguas, pero la falta de ellos, como ya se ha comentado, nos llevó a tomar nosotros mismos los datos para realizar un caso lo más real posible con el modelo construido.

5

Apéndices

Tal y como se ha nombrado anteriormente los modelos han sido programados en XPRESS, la programación de los mismos se aportará en estos apéndices.

5.1. Apéndice 1: Modelo de una persona aleatorio.

```
model "Una persona"
uses "mmxprs","mmive","mmsystem";

!optional parameters section
parameters
n = 10
end-parameters

!forward procedure write_results
forward procedure draw
forward procedure write_results
!sample declarations section
declarations

V = 0..n

! Constants
MINDIST = 0
MAXDIST = 30
FACTOR = 2.2
FACTORHOUSES = 415

coord_x: array(V) of integer
```

```
coord_y: array(V) of integer

COST: array(V,V) of real
HOUSES: array(V,V) of integer
near: array(V) of integer
l: array(V,V) of integer
x: array(V,V) of mpvar
  y: array(V) of mpvar
f: array(V,V) of mpvar

end-declarations

! Fill in with random locations
setrandseed(5)

forall(i in V) do
coord_x(i):=round(random*100)
coord_y(i):=round(random*100)
end-do

!We calulate the cost-matrix
forall(i,j in V | i<j)
COST(i,j):= round( abs(coord_x(i)-coord_x(j)) +
              abs(coord_y(i)-coord_y(j)) )
forall(i,j in V | i<j) COST(j,i):=COST(i,j)

forall(i in V) do
near(i) := 0;
forall(j in V | j <> i) do
if(COST(i,j) > MINDIST and COST(i,j) < MAXDIST) then
near(i) += 1
end-if
end-do
end-do

forall(i,j in V | i < j) do
l(i,j) := 0
l(j,i) := 0
if(COST(i,j) > MINDIST and COST(i,j) < MAXDIST) then
if (random < FACTOR / near(i) ) then
l(i,j) := 1
l(j,i) := 1
end-if
```

```
end-if
end-do

forall(i,j in V | i < j) do
if(l(i,j)=1) then
HOUSES (i,j):= round(COST(i,j)/10 +1)
HOUSES (j,i):= HOUSES (i,j)
else
HOUSES (i,j):=0
HOUSES (j,i):=0
end-if
end-do

!Objetive function
TotalCost:= (sum(i,j in V |i<>j)COST(i,j)*x(i,j)) +
            (sum(i,j in V |i<j) (FACTORHOUSES*HOUSES(i,j)*l(i,j)))

!Constraints
forall(i in V) sum(j in V | j<>i) x(i,j) = y(i)
forall(i in V) sum(j in V | j<>i) x(j,i) = y(i)
forall(i,j in V | i<j) x(i,j)+x(j,i)>=l(i,j)
y(0)= 1
forall(i in V | i<>0) (sum(j in V | j<>i) f(i,j) -
                       sum(j in V | j <> i) f(j,i)) = y(i)
forall(i,j in V| j<>i) f(i,j) <= 2*n*x(i,j)
forall (i,j in V) x(i,j) is_binary
forall (i in V) y(i) is_integer
forall(i,j in V | j<>i) f(i,j) >= 0

minimize(TotalCost)

writeln("Solving with n=", n," corners.")

draw
write_results
procedure draw
IVEerase
    IVEzoom(-5,-5,100+5,100+5)
    corners:=IVEaddplot("corners",IVE_RED)
    route1:=IVEaddplot("route1",IVE_BLUE)

forall(i in V) do
IVEdrawlabel(corners,coord_x(i),coord_y(i),""+i)
```

```
end-do
!draw links
forall(i,j in V) do

      if getsol(x(i,j)) > 0.99  then
   IVEdrawarrow(route1,coord_x(i),coord_y(i),
             coord_x(j),coord_y(j))
   end-if
end-do
end-procedure

procedure write_results
forall( i,j in V| i<>j) do
if getsol(x(i,j)) > 0.99 then
writeln("x(",i,",",j,")= ", getsol(x(i,j)),"\tl(",i,",",j,")= ",
 l(i,j),"\tc(",i,",",j,")= ",COST(i,j), "+",
 (FACTORHOUSES*HOUSES(i,j)*l(i,j)))
end-if
end-do
writeln("Optimal cost  : ", getobjval)
fixCost := sum(i,j in V |i<j) (COST(i,j)*l(i,j) +
 FACTORHOUSES*HOUSES(i,j)*l(i,j))
writeln("Fix cost      : ", fixCost)
writeln("total l: ", sum(i,j in V |i<j)l(i,j))
writeln("Variable  cost: ", getobjval - fixCost)
end-procedure

end-model
```

5.2. Apéndice 2: Modelo de k personas aleatorio.

```
model "RUTAS"
uses "mmxprs","mmive","mmsystem";

!optional parameters section
parameters
n =12
p = 2
end-parameters

forward procedure draw
forward procedure write_results
```

```
!sample declarations section
declarations

V = 0..n
P = 1..p

! Constants
MINDIST = 0
MAXDIST = 50
FACTOR = 2.2
MAXDIFF = 2000
FACTORHOUSES = 415

coord_x: array(V) of integer
coord_y: array(V) of integer

COST: array(V,V) of real
HOUSES: array(V,V) of integer
near: array(V) of integer
l: array(V,V) of integer
x: array(V,V,P) of mpvar
    y: array(V,P) of mpvar
f: array(V,V,P) of mpvar
z: array(V,V,P) of mpvar
c: array(P) of mpvar

end-declarations

! Fill in with random locations
setrandseed(1)

forall(i in V) do
coord_x(i):=round(random*100)
coord_y(i):=round(random*100)
end-do

!We calulate the cost-matrix
forall(i,j in V | i<j) COST(i,j):= round( abs(coord_x(i)-coord_x(j))
                                        + abs(coord_y(i)-coord_y(j)) )
forall(i,j in V | i<j) COST(j,i):=COST(i,j)

!We calulate the cost-matrix
forall(i,j in V | i<j) COST(i,j):= round( abs(coord_x(i)-coord_x(j))
```

```
                                        + abs(coord_y(i)-coord_y(j)) )
forall(i,j in V | i<j) COST(j,i):=COST(i,j)

forall(i in V) do
near(i) := 0;
forall(j in V | j <> i) do
if(COST(i,j) > MINDIST and COST(i,j) < MAXDIST) then
near(i) += 1
end-if
end-do
end-do

forall(i,j in V | i < j) do
l(i,j) := 0
l(j,i) := 0
if(COST(i,j) > MINDIST and COST(i,j) < MAXDIST) then
if (random < FACTOR / near(i) ) then
l(i,j) := 1
l(j,i) := 1
end-if
end-if
end-do

forall(i,j in V | i < j) do
if(l(i,j)=1) then
HOUSES (i,j):= round(COST(i,j)/10 +1)
HOUSES (j,i):= HOUSES (i,j)
else
HOUSES (i,j):=0
HOUSES (j,i):=0
end-if
end-do


K := round ((sum(i,j in V | i < j) l(i,j)) * 3 / p)


forall ( k in P) do

c(k)=(sum(i,j in V |i<>j) COST(i,j)*x(i,j,k)) +
     (sum(i,j in V |i<>j) (FACTORHOUSES*HOUSES(i,j)*z(i,j,k)))

end-do
```

```
!Objetive function
TotalCost:= sum(i,j in V, k in P| i<>j) COST(i,j)*x(i,j,k)+
           (sum(i,j in V |i<j) (FACTORHOUSES*HOUSES(i,j)*l(i,j)))

!Constraints
forall(i in V, k in P) sum(j in V | j<>i) x(i,j,k) = y(i,k)
forall(i in V, k in P) sum(j in V | j<>i) x(j,i,k) = y(i,k)

forall(i,j in V | i<j) sum(k in P)(z(i,j,k)+z(j,i,k))=l(i,j)

forall(k in P) y(0,k)= 1

forall(i in V, k in P | i<>0) sum(j in V | j<>i) f(i,j,k) -
                              sum(j in V | j <> i) f(j,i,k) = y(i,k)
forall(i,j in V, k in P | j<>i) f(i,j,k) <= K * x(i,j,k)

forall (i,j in V, k in P) x(i,j,k) is_binary
forall (i,j in V, k in P) z(i,j,k) is_binary
forall (i in V, k in P) y(i,k) is_integer
forall(i,j in V, k in P | j<>i) f(i,j,k) >= 0
forall (k in P) c(k) >=0

!Compensar las lecturas
forall(k1,k2 in P | k1<>k2) (c(k1)-c(k2)) <= MAXDIFF
forall(i,j in V, k in P) x(i,j,k) >= z(i,j,k)

minimize(TotalCost)

writeln("Solving with n=", n," corners.")

draw
write_results
procedure draw
IVEerase
    IVEzoom(-5,-5,100+5,100+5)
    corners:=IVEaddplot("corners",IVE_RED)
    route1:=IVEaddplot("route1",IVE_BLUE)
    route2:=IVEaddplot("route2",IVE_GREEN)
    route3:=IVEaddplot("route3",IVE_MAGENTA)

forall(i in V) do
IVEdrawlabel(corners,coord_x(i),coord_y(i),""+i)
```

```
end-do
!draw links
forall(i,j in V, k in P) do
if(k = 1) then
     if getsol(x(i,j,k)) > 0.99  then
       IVEdrawarrow(route1,coord_x(i),
             coord_y(i),coord_x(j),coord_y(j))
     end-if
     elif(k=2) then
     if getsol(x(i,j,k)) > 0.99  then
       IVEdrawarrow(route2,coord_x(i),
             coord_y(i),coord_x(j),coord_y(j))
     end-if
     else
     if getsol(x(i,j,k)) > 0.99  then
       IVEdrawarrow(route3,coord_x(i),
             coord_y(i),coord_x(j),coord_y(j))
     end-if
     end-if
end-do
end-procedure

procedure write_results
forall (k in P, i, j in V | i<>j) do
if (getsol(x(i,j,k)) > 0.1) then
write( "x(", i, ",", j,",", k, ")= ", getsol(x(i,j,k)))
write( "\tz(", i, ",", j,",", k, ")= ", getsol(z(i,j,k)))
write( "\tl(", i, ",", j,")= ", l(i,j))
writeln( "\tc(", i, ",", j,",", k, ")= ",
         COST(i,j)*getsol(x(i,j,k)), " + ",
 FACTORHOUSES*HOUSES(i,j)*getsol(z(i,j,k)) )
end-if
end-do
forall(k in P) do
writeln("c(",k,")= ",(sum(i,j in V |i<>j) COST(i,j)*getsol(x(i,j,k))),
  " + " , (sum(i,j in V|i<>j ) (FACTORHOUSES*HOUSES(i,j) *
  getsol(z(i,j,k)))), " = ",     getsol(c(k)) )
end-do
writeln("Optimal cost: ", getobjval )
end-procedure

end-model
```

5.3. Apéndice 3: Modelo por zonas aleatorio.

```
model "RUTASZONAS"
uses "mmxprs","mmive","mmsystem";

!optional parameters section
parameters
n = 11
Q= 35000 !cantidad de tiempo
k = 3
end-parameters

forward procedure draw
forward procedure write_results

declarations

V = 0..n

coord_x: array(V) of integer
coord_y: array(V) of integer
COST: array(V,V) of real
x: array(V,V) of mpvar
  u: array(V) of mpvar
d: array(V) of integer
end-declarations


! Fill in with random locations
setrandseed(1)

forall(i in V) do
coord_x(i):=round(random*100)
coord_y(i):=round(random*100)
end-do

!We calulate the cost-matrix
forall(i,j in V | i<j) COST(i,j):= round( abs(coord_x(i)-coord_x(j))
                                        + abs(coord_y(i)-coord_y(j)) )
forall(i,j in V | i<j) COST(j,i):=COST(i,j)

forall ( i in V) do
d(i):= 10 + round(random*20)
```

```
end-do

!Objetive function
TotalCost:= sum(i,j in V| i<>j) COST(i,j)*x(i,j)

!Constraints
forall(j in V| j<>0) sum(i in V | j<>i) x(i,j) = 1
forall(i in V| i<>0) sum(j in V | j<>i) x(i,j) = 1
sum(i in V| i<>0 ) x(i,0) = sum(j in V| j<>0 ) x(0,j)
forall(i in V| i<>0) sum(j in V| j<>i ) f(i,j) -
                     sum(j in V| j<>i ) f(j,i) = d(i)
forall(i,j in V | j<>i )  f(i,j) <= Q * x(i,j)
forall(i,j in V | j<>i )  f(i,j) >= 0

forall (i,j in V) x(i,j) is_binary

minimize(TotalCost)

writeln("Solving with n=", n," corners.")

writeln("The obligatories arcs are")

draw
write_results
procedure draw
IVEerase
    IVEzoom(-5,-5,100+5,100+5)
    corners:=IVEaddplot("corners",IVE_RED)
    route1:=IVEaddplot("route1",IVE_BLUE)
    !route2:=IVEaddplot("route2",IVE_GREEN)
    !route3:=IVEaddplot("route3",IVE_MAGENTA)

forall(i in V) do
IVEdrawlabel(corners,coord_x(i),coord_y(i),""+i)
end-do
!draw links
forall(i,j in V | i<>j) do
if getsol(x(i,j)) > 0.99 then
IVEdrawarrow(route1,coord_x(i),coord_y(i),coord_x(j),coord_y(j))
end-if
end-do
end-procedure
```

```
procedure write_results
writeln("Costo total= ", getobjval+ sum(i in V) d(i) )
writeln("N° de personas= ", sum (j in V | j <> 0) getsol(x(0,j)) )
forall( i,j in V| i<>j) do
if getsol(x(i,j)) > 0.99 then
write("x(",i,",",j,")= ", getsol(x(i,j)), "    f(",i,",",j,")= ",
       getsol(f(i,j))  )
writeln("   c(",i,",",j,")= ", COST(i,j), "    d(j)= ", d(j) )
end-if
end-do
writeln("Costo recorrido= ", getobjval)
writeln("Costo zonas= ", sum(i in V) d(i))

end-procedure

end-model
```

5.4. Apéndice 4: Modelo de una persona real.

En este apéndice mostraremos únicamente cómo se leerían los datos de un fichero ya que es la única diferencia con respecto con el caso aletorio. Se sustituiría la generación de datos aleatorios por:

```
parameters
n = 182
File="zona11.txt" !Este es el único parámetro
                  !diferente respecto al caso aleatorio.
end-parameters

fopen(File,F_INPUT)
readln(narcs)
forall (a in 1..narcs) do
readln(index_i, index_j, temp_cost, temp_h)
if(a = 1) then
first_index := index_i
end-if
COST(index_i, index_j) := temp_cost
COST(index_j, index_i) := temp_cost
HOUSES(index_i, index_j) := temp_h
HOUSES(index_j, index_i) := temp_h
if (temp_h > 0) then
l(index_i, index_j) := 1
```

```
l(index_j, index_i) := 1
else
l(index_i, index_j) := 0
l(index_j, index_i) := 0
end-if
end-do
fclose(F_INPUT)
```

5.5. Apéndice 5: Modelo de zonas real.

Al igual que en el apéndice anterior solo mostraremos las diferencias con el aleatorio.

```
parameters
n = 11
Q= 35000 !cantidad de tiempo
nzonas =11
File1="pesodezona.txt"
File2="distzonas.txt"
end-parameters

fopen(File1,F_INPUT)
forall (a in 1..nzonas) do
readln(index_i,temp_d)
d(index_i) := temp_d
end-do
fclose(F_INPUT)

fopen(File2,F_INPUT)
forall (a in 1..66) do
readln(index_i, index_j,temp_cost)
COST(index_i, index_j) := temp_cost
COST(index_j, index_i) := temp_cost
end-do
fclose(F_INPUT)
```

Bibliografía

[1] D. Davendra. 2010. *"Traveling salesman problem, theory and applications". InTech.*

[2] M. M. Flood. 1956. *"The traveling-salesman problem. Operations Research", 4:61-75.*

[3] G. Gutin and A. P. Punnen. 2002. *"The Traveling Salesman Problem and its variations".*

[4] Á. Corberán and G. Laporte. 2013. *"Arc Routing: Problems, Methods, and Applications", MOS-SIAM Series on Optimization.*

[5] P. Toth and D. Vigo. 2014. *"Vehicle routing: Problems, Methods, and Applications". Second Edition, MOS-SIAM Series on Optimization.*

[6] FICO^{TM} Xpress Optimization Suite. 2013. *"Xpress-Mosel (User guide)".*

Lista de Tablas

Lista de Figuras

A routing problem facing a public water supply company

Serezade Tedauit González Torres
Facultad de Ciencias · Sección de Matemáticas
Universidad de La Laguna
alu0100828900@ull.edu.es

Abstract

Routing problems are one of the most outstanding branches of Operational Research. These problems arise naturally in a large variety and number of areas. Taking this into account, the following report exposes a routing problem that is born in a public water supply company, aiming to optimize the routes made by the workers in charge of reading meters. To carry out the approach of this problem, a study of the main problems of routes will be shown briefly: travelling salesman problem (TSP) [1, 2, 3], Chinese postman problem [4] and vehicle routing problem (VRP) [5]. This will facilitate the formulation of the model carried out. This model has matured developed until obtaining a model that reflects the real problem as much as possible and solves it in a computationally viable way.

1. Routing problems introduction

Operational Research is a modern science that uses mathematical models, statistics and algorithms to formulate and solve complex problems that give us a well structured vision of reality, thus obtaining optimal solutions to problems and facilitating decision making. It is important to point out that the advance of this science is linked to the computer advance, since this has made it possible to solve problems in practice and obtain solutions that would otherwise involve a huge calculation time making them unfeasible.
This branch of mathematics is used, among other, for the routing optimization. In this work we will try to carry out a routing optimization developing a mathematical model with which to obtain the best route that a meter reader of a public water supply company should complete.

2. Problem formulation

We will develop the models that solve the different variants of our problem, which consist in reading the water meters in an optimal way. For this, the following has been considered: each street of the route to be optimized has been separated into two, with the ends being a node side and being connected to each other by two arcs.

The variants of the problem that have been raised are:

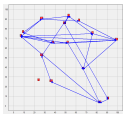
- **A person** is responsible for leaving the deposit (the company) and going through the mandatory streets (and others if necessary) to perform the reading of counters in the shortest time possible.
- **Several persons** leave the company to complete their workday, reading as many counters as possible.
- **The problem of zones** arises due to the little power of the problem of k people, and to simulate the reality more accurately than the model with just one person. This variant is carried out in two phases, the first consists of dividing the area to be studied in several (relatively small) areas and applying the model of one person to each of them, in this way we get the cost of each zone. In the second phase, using the results of the first, a model based on the CVRP is applied, to get the routes between zones and the number of people needed to travel them all in a certain time (their working day).

3. Models aplication

Each of the named variants was programmed in the XPRESS mathematical programming environment [6]. First it was done by randomly generating data and then applied to a real simulation.

- **Random data**

One person and 15 nodes:



- **Map of the real case divided by zones**

That it has to be covered by four people in the following way:

References

[1] D. Davendra. 2010. *"Traveling salesman problem, theory and applications". InTech.*

[2] M. M. Flood. 1956. *"The traveling-salesman problem. Operations Research", 4:61-75.*

[3] G. Gutin and A. P. Punnen. 2002. *"The Traveling Salesman Problem and its variations".*

[4] Á. Corberán and G. Laporte. 2013. *"Arc Routing: Problems, Methods, and Applications", MOS-SIAM Series on Optimization.*

[5] P. Toth and D. Vigo. 2014. *"Vehicle routing: Problems, Methods, and Applications". Second Edition, MOS-SIAM Series on Optimization.*

[6] FICO™ Xpress Optimization Suite. 2013. *"Xpress-Mosel (User guide)".*