Optimización de rutas de recogida de residuos en zonas mixtas urbana-rurales y orografía singular

Routing Optimization for waste collection in urban-rural mixed areas and singular orography

Tania Yurena Afonso LLorente
Departamento de Ingeniería Informática y de Sistemas
Escuela Técnica Superior de Ingeniería Informática
Trabajo de Fin de Grado

La Laguna, 09 de junio de 2014

D. **Julio Brito Santana**, con N.I.F. 42.812.193-Q profesor Titular de la Escuela Técnica Superior de Ingeniería Informática adscrito al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna

C E R T I F I C A

Que la presente memoria titulada:

"Optimización de rutas de recogida de residuos en zonas mixtas urbana-rurales y orografía singular."

ha sido realizada bajo su dirección por D. Tania Yurena Afonso LLorente, con N.I.F. 78.721.696-X.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna, a 9 de junio de 2014

Agradecimientos

Quisiera agradecer a mi tutor D. Julio Brito Santana su apoyo constante y la gran confianza que ha depositado en mí a lo largo de todas las tareas que bajo su tutela he realizado, contando siempre con su inestimable consejo y experiencia.

A D. Dagoberto Castellanos Nieves, quisiera agradecerle su estrecha colaboración en muchas de las tareas realizadas, así como su completa disposición a resolver cualquier duda o problema que se haya podido plantear.

Finalmente muchas gracias a mis padres, por apoyarme en todas las decisiones que he tomado a lo largo de la vida, hayan sido buenas o malas. A ellos no sólo les debo este proyecto, sino la realización de la titulación completa.

Resumen

Este proyecto tiene como finalidad el modelado y resolución de un problema de planificación óptima de rutas de recogida de residuos en municipios que combinan zonas rurales y urbanas. Este problema tiene peculiares restricciones dadas por, la combinación de zonas de concentración y dispersión de población, y características orográficas singulares que determinan la orientación de las rutas. Una vez modelado se diseña e implementa un método heurístico que permite encontrar soluciones a este problema. El modelo que responde al problema estudiado es el Problema de Rutas de Vehículos con Arcos Capacitados. El procedimiento metaheurístico implementado es una variante del Procedimiento de Búsqueda Voraz Adaptativo Probabilista.

Palabras clave

Gestión de residuos, metaheurística, optimización combinatoria, Problema de Rutas de Vehículos

Abstract

This project aims to modeling and resolve a problem of routing optimization for waste collection in urban-rural mixed areas. This problem has peculiar constraints combination of areas of concentration and dispersion and unique orographic characteristics that determine the orientation of the routes. Once modeled a heuristic method will be defined and implemented that allows to find solutions. Through the investigation of the vehicle routing problem we can conclude that the Capacitated Arc Routing Problem fits better to our problem. We made an implementation of the metaheuristic algorithm Greedy Randomized Adaptative Search Procedure.

Keywords

Waste management, metaheuristic, combinatorial optimization, Vehicle Routing Problem

Índice general

Índice de Figuras

Índice de Tablas

Tabla 23: Pruebas del fichero de entrada ejemplo con diez mil iteraciones.

Tabla 24: Pruebas del fichero de entrada ejemplo con cien mil iteraciones.

CAPÍTULO 1

1. INTRODUCCIÓN

En este capítulo describe una introducción al problema de rutas de vehículos propuesto, incluyendo la motivación del mismo, su aplicación y las técnicas usadas para alcanzar soluciones al mismo. Además en esta introducción se declaran los objetivos de este proyecto.

Este proyecto surgió de la solicitud por parte del ayuntamiento de Icod de los Vinos de un software que les ayudase en su labor de la recogida de residuos urbanos. En dicho municipio actualmente no hay una gestión de residuos completamente implementada, sino un sistema de recogida de residuos puerta a puerta. Este municipio de tipo rural dispone de una flota de vehículos con los que recorren todas las calles para la recogida de residuos ante la carencia de contenedores en las calles. Dichos vehículos se encuentran con grandes dificultades en su labor, ya que el municipio tiene una orografía muy compleja, con grandes pendientes en todo el municipio, por lo que tienen que comenzar en la parte más alta para poder ir cargando residuos a medida que descienden hasta el depósito. Además, muchas de las calles son demasiado estrechas y no pueden pasar, o bien no tienen salida. A continuación se muestra una imagen de Icod de los Vinos. En ella se puede observar la compleja orografía del terreno.

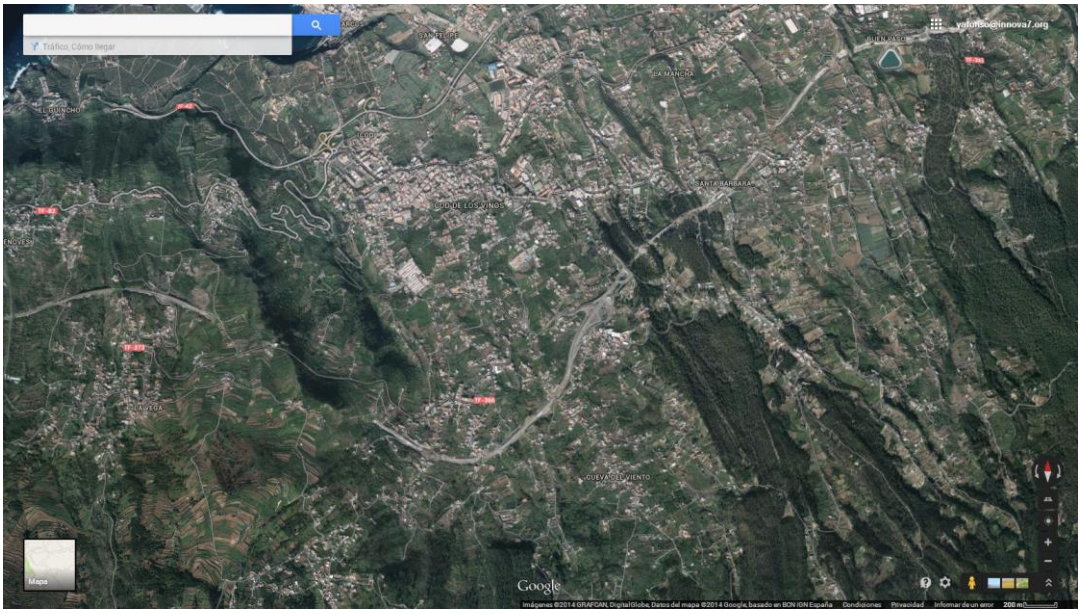

Figura 1: Imagen de Icod de los Vinos.

Este problema fue propuesto por la empresa pública que gestiona la recogida de residuos en el municipio de Icod de los Vinos. Las dificultades a la hora de disponer de los datos correspondientes, en los formatos adecuados impidió resolver el problema específico de dicho municipio. Para resolver el problema se tomó como referencia una instancia semejante con los mismos objetivos y restricciones.

Una de las partes fundamentales en el ámbito de la gestión urbanística hoy en día es la recogida y gestión de residuos. Todas las personas generamos gran cantidad de residuos tanto orgánicos como reciclables y es necesaria una completa gestión de dichos residuos, ya que en concreto los residuos orgánicos son los más reciclables. Con una buena gestión se pueden utilizar como enmienda orgánica, fertilizando en viveros, sustrato para macetas, mejora del suelo agrícola,... Pero esta gestión no está totalmente centralizada. Hay gran cantidad de reglamentos y leyes que describen cómo debería de hacerse la gestión a grandes rasgos, incluyendo características concretas que deberían de tener los sistemas sobre todo en función de la población que haya en el lugar en cuestión. De dicha gestión de residuos normalmente se encargan los ayuntamientos de cada municipio, como es el caso de las Islas Canarias. Cada uno de sus municipios tiene unos determinados convenios con empresas que gestionan sus residuos, o bien tienen sus propios depósitos y vehículos para realizar las labores de recogida y gestión. El problema que genera esta vaga descripción de las obligaciones con respecto a la recogida de residuos es que hay municipios no le están prestando la debida atención, llegando incluso a carecer de estos servicios.

A raíz de este problema, surgen gran cantidad de alternativas que se pueden aplicar para optimizar de unas u otras formas esta planificación de la recogida de residuos.

Por ejemplo, se puede tener en cuenta el coste que supone tener más o menos vehículos para realizar la recogida de residuos, en función de si conviene realizar las labores en un determinado tiempo, de la distancia a recorrer por cada vehículo, del coste que supone mantener dichos vehículos,… Con el fin de reducir en la medida de lo posible todos estos inconvenientes, se busca optimizar en todo lo posible en ese campo. Es muy importante tratar de buscar qué requerimientos es conveniente tener en cuenta para cada problema particular.

En el caso de estudio de este proyecto, se dan una serie de situaciones que dificultan en gran medida la recogida de residuos. Para comenzar, la orografía de las islas es muy complicada, ya que al ser de origen volcánico hay gran cantidad de desniveles y barrancos que provocan dificultosos recorridos necesarios para cubrir toda la recogida de residuos. Por otro lado, hay un gran desnivel en el terreno, esto es, en muy corta distancia hay una gran elevación del terreno, provocada por dicho origen volcánico. Todos los municipios tienen una gran pendiente, así que los camiones de recogida tienen que buscar formas que les permitan realizar su trabajo, comenzar a recoger los residuos por las partes más altas, evitar subir con carga en la medida de lo posible bajo las condiciones y restricciones específicas impuestas por la orografía.

A través de este proyecto se busca investigar en el ámbito de la recogida de residuos en zonas mixtas urbanas y rurales con esta orografía singular. Para este tipo de problemas es necesario aplicar técnicas heurísticas debido a la complejidad que conllevan y porque no existen soluciones exactas o no se obtienen en un tiempo razonable. En definitiva, el objetivo principal es mejorar la planificación de rutas para reducir los costes ya sean de distancia, ahorrar en tiempo de trabajo, en distancia recorrida,…

La raíz del problema de este proyecto surge de una problemática que sucede en la vida real. Actualmente hay municipios que se encuentran sin una recogida de residuos completamente implementada, no hay puntos concretos de recogida de residuos a los que los camiones tengan que ir a recoger, sino que las personas dejan la basura en las propias calles. Por esa razón, en vez de tener que realizar rutas para los camiones de forma que tengan que recorrer todos los puntos, tienen que pasar por todas y cada una de las calles donde hay viviendas para recoger la basura. Esto complica en gran medida el problema, además de incrementar la complejidad del propio trabajo. Hay que recorrer demasiada más distancia, y hay calles a las que los camiones no pueden acceder en la vida real.

Muchos problemas de decisión existentes en el mundo real, en particular aquellos relacionados con la producción, rutas de vehículos, logística, planificación,… pueden ser formulados como problemas de optimización. En el problema tratado nos encontramos con un problema de Optimización combinatoria, una clase dentro de los problemas de optimización. En estos problemas, las soluciones son consideradas variables discretas y el proceso de búsqueda de soluciones consiste en explorar el

espacio de soluciones del problema representado mediante listas, conjuntos matrices o grafos. La Optimización combinatoria va adquiriendo un creciente interés debido a las soluciones alcanzadas sobre diferentes problemas reales así como el avance en el estudio teórico-práctico en diferentes áreas dentro de la Investigación Operativa.

Dentro de los problemas de optimización combinatoria, se encuentran los de rutas de vehículos. El Problema de Rutas de Vehículos (Vehicle Routing Problem o VRP) consiste en atender demandas sobre ciertas calles de una red vial a través de una flota homogénea de vehículos, los cuales inician y finalizan sus recorridos en un depósito [1]. El objetivo del problema consiste en minimizar el costo total del recorrido de manera que se atiendan todas las demandas solicitadas sin sobrecargar la capacidad de carga de los vehículos utilizados. El Problema de Rutas de Arcos Capacitados (Capacitated Arc Routing Problem o CARP) consiste en una flota de vehículos, con una capacidad conocida (en términos de tiempo o volumen, o ambos), debe recorrer los arcos de un grafo, con coste total mínimo y cumpliendo los límites de capacidad de los vehículos. La diferencia entre ambos problemas radica en que el primero hay un conjunto de nodos que se deben visitar y en el segundo se trata de arcos.

Uno de los beneficios del estudio de este problema radica en su aplicación real y concreta para resolver problemas de organismos públicos y empresas. Tanto los organismos públicos como las empresas deben obtener soluciones rápidas y con el menor uso posible de recursos. Entre las aplicaciones del CARP o de sus extensiones relacionadas con servicios públicos podemos mencionar la gestión de la recogida de residuos.

El problema planteado es NP-Hard, esto es que no existe un algoritmo polinómico para su para su solución, y por tanto no puede ser comprobado en un tiempo razonable.

La baja calidad de soluciones devueltas por las heurísticas tradicionales, motivó el uso y desarrollo de nuevas propuestas algorítmicas [2]. Las metaheurísticas, definidas como técnicas generales usadas para guiar o controlar un método heurístico y específico del problema con el objetivo de mejorar el rendimiento o robustez de los métodos subordinados. Estas han sido aplicadas a diferentes problemas de optimización, entre ellos CARP, gracias a su versatilidad para adaptarse a nuevos problemas, como también su sencillez de implementación, calidad de las soluciones logradas y el bajo tiempo computacional asociado.

A lo largo de este documento se irán describiendo cada uno de los pasos que se han seguido para el proceso de investigación y desarrollo. Para comenzar se describe cómo funciona la gestión de residuos, haciendo referencia a la legislación vigente que han de cumplir cada uno de los municipios, especificando también a la normativa actual aplicada en las Islas Canarias. A continuación se describen los modelos, comenzando por el Problema del Viajante de Comercio, problema base en la

optimización de rutas de vehículos en la que un solo vehículo ha de recorrer todos los puntos. Después se describe el Vehicle Routing Problem, generalización del Problema del Viajante de Comercio en el que hay una flota de vehículos, cada uno con su ruta, y en el TSP sólo hay una ruta de un vehículo. Posteriormente se describe el Problema del Cartero Chino (Chinese Postman Problem o CPP), ya que en el problema que atañe a este proyecto se busca recorre todas las calles, no vértices concretos, por lo que el Problema del Cartero Chino se antoja muy similar. Por último se hace referencia al Capacitated Arc Routing Problem, generalización del Problema del Cartero Chino en el que hay que recorrer todos los arcos pero es posible hacerlo con más de un vehículo.

En la siguiente sección del proyecto se describe la implementación de la metaheurística Procedimiento de Búsqueda Voraz Adaptativo Probabilista. Se ha buscado la forma en que esta metaheurística resuelva el problema de rutas de vehículos con arcos capacitados, de forma que se optimicen ciertas restricciones particulares de este problema real, el de recogida de residuos en zonas rurales urbanas-mixtas con orografía singular. Estas restricciones son la distancia que ha de recorrer cada vehículo, el coste que supone hacer ese recorrido, el tiempo invertido para recorrer todas las rutas,… En la última parte de este documento se muestran los resultados obtenidos tras la experimentación realizada. A continuación se describe la estructura de los contenidos de la memoria:

- **Capítulo 1**
  En este capítulo describe una introducción al problema de rutas de vehículos propuesto, incluyendo la motivación del mismo, su aplicación y las técnicas usadas para alcanzar soluciones al mismo. Además en esta introducción se declaran los objetivos de este proyecto.

- **Capítulo 2**
  En este capítulo se busca mostrar la problemática de la gestión de residuos en toda su extensión. Comienza con una descripción general acerca de la gestión de residuos, además de información legal relacionada con este ámbito. A lo largo de este capítulo también se habla de la gestión de residuos en las Islas Canarias.

- **Capítulo 3**
  Desde este capítulo se busca ilustrar acerca de los diferentes modelos existentes que estudian este tipo de problemas, exponiendo la relación entre cada uno de ellos. Se incluye también el modelo matemático de cada uno de ellos.

- **Capítulo 4**
  A través de este cuarto capítulo se realiza un estudio de varios de los métodos existentes, haciendo mayor hincapié en los que finalmente se ha decidido utilizar.

- **Capítulo 5**
  En este capítulo se realiza un análisis del problema, que incluye qué sucede en los problemas reales de recogida de residuos y los detalles específicos del problema. A continuación se describen los procedimientos que se van a utilizar para llevar a cabo este proyecto, incluyendo la función de costes que se va a aplicar.

- **Capítulo 6**
  En este capítulo se describe cómo ha sido el desarrollo y la implementación del algoritmo Procedimiento de Búsqueda Voraz Adaptativo Probabilista. Esta descripción abarca desde el comienzo de la implementación hasta la descripción de cada una de las clases del programa. Además, en un segundo apartado se describen los métodos de optimización utilizados.

- **Capítulo 7**
  En este capítulo se describe la experimentación elaborada. Para la realización de estas pruebas hemos utilizado una instancia que ejemplifica las características del problema real de Icod de los Vinos.

- **Conclusiones**
  Resumen de las conclusiones tanto en español (capítulo 8) como en inglés (capítulo 9).

- **Bibliografía**
  Recopilación de referencias bibliográficas que han sido útiles para el desarrollo de este proyecto.

- **Anexo I**
  Muestra de los datos referidos en el capítulo 7 de experimentación.

CAPÍTULO 2

2. GESTIÓN DE RESIDUOS

En este capítulo se busca mostrar la problemática de la gestión de residuos en toda su extensión. Comienza con una descripción general acerca de la gestión de residuos, además de información legal relacionada con este ámbito. A lo largo de este capítulo también se habla de la gestión de residuos en las Islas Canarias. También se trata la organización de la recogida de residuos.

Se denomina gestión de residuos al control y manejo de todo el ciclo de los residuos domiciliarios, industriales, hospitalarios y agropecuarios. Este ciclo está formado por: recolección, transporte, procesamiento, tratamiento, reciclaje y transferencia hasta el depósito final. Los residuos son producidos por las actividades humanas. Esto produce efectos perjudiciales en la salud, en el medio ambiente y en la estética del entorno.

Los residuos se clasifican en domiciliarios, industriales, hospitalarios y agropecuarios. Cada uno de ellos se gestiona de manera diferente. Además, la gestión de residuos abarca la gestión de residuos peligrosos, que son los residuos químicos o físicos (radiactivos) muy perjudiciales para la salud humana.

La recogida de los residuos urbanos consiste en su recolección para efectuar su traslado a las plantas de tratamiento. Básicamente existen dos tipos fundamentales de recogida: la recogida no selectiva y la recogida selectiva.

En la recogida no selectiva, los residuos se depositan mezclados en los contenedores, sin ningún tipo de separación. Este tipo de recogida ha sido la habitual hasta hace algunos años, y sigue siéndolo en algunos municipios rurales o de menor población.

La recogida selectiva consiste en agrupar y clasificar los residuos de acuerdo con sus características y propiedades, con el fin de facilitar posteriormente su tratamiento. Se basa en que los ciudadanos realicen una selección de los productos recuperables, colocándolos en recipientes independientes. Esta selección de residuos puede hacerse depositando cada material en su contenedor correspondiente, o bien realizando una recogida puerta a puerta. La recogida puerta a puerta es un sistema para recoger residuos de manera selectiva cuya principal característica es que cada vivienda recoge de manera separada los residuos que produce y cada día de la semana

depositan delante de sus casas un tipo de residuo determinado. En este sistema se establece un horario con el fin de que los residuos permanezcan el menor tiempo posible en la vía pública, habiendo siempre posibilidad de dejar los residuos en los denominados puntos de emergencia para gente que no pueda adaptarse al horario del puerta a puerta, que son contenedores para que la gente deposite en cualquier momento sus residuos separados.

La recogida selectiva y el reciclaje permiten ahorrar recursos escasos y parte de la energía necesaria para la fabricación de los productos a partir de materias primas vírgenes.

Los residuos constituyen uno de los problemas ambientales más graves de las sociedades modernas, en particular de las más avanzadas e industrializadas. Se trata de un problema de aumento, que no deja de agravarse debido al creciente volumen generado y a la estrecha relación de paralelismo entre los niveles de renta y de calidad de vida, y el volumen de residuos que generamos. A la vista de tal incremento en la generación de residuos, se llega a la conclusión de la necesidad de regularlos con rigor ecológico y planificar la puesta en práctica de esa regulación de manera racional y realista.

A través de la directiva marco europea de residuos (directiva 2006/12/CE) se señala la necesidad de que los Estados miembros dispongan de instrumentos jurídicos y de planificación sobre residuos. En la Estrategia Comunitaria de gestión de residuos, adoptada por resolución del Consejo del 24 de Febrero de 1997, se contempla la conveniencia de elaborar Planes de residuos. Esta prioridad también es recogida en la "Estrategia Temática de Prevención y Reciclaje de Residuos", en la que se contempla, incluso, la elaboración de planes específicos de prevención de residuos.

En el Boletín Oficial del Estado con fecha del 22 de Abril de 1998 se publicó la Ley de Residuos (Ley 10/1998, del 21 de Abril), siendo esta aplicable a cualquier clase de residuo con la excepción de las emisiones atmosféricas, residuos radioactivos, vertidos efluentes líquidos a las aguas continentales, vertidos desde la tierra al mar y vertidos provocados por buques y aeronaves al mar. La creación de esta Ley de Residuos en el Estado Español fue provocada por una política de residuos adoptada por parte de la Unión Europea, consistente en abandonar la clasificación centrada en dos únicas modalidades (residuos generales y residuos peligrosos) y pasar al establecimiento de una norma común para todos ellos.

En la Ley 10/1998, de Residuos, artículos 5 y 6, se establece la obligación de elaborar y aprobar Planes Nacionales de Residuos, que se confeccionarán por integración de los respectivos Planes autonómicos. En los Planes Nacionales deben figurar objetivos de reducción, reutilización, reciclaje, otras formas de valorización, y eliminación, así como los medios para conseguirlos, el sistema de financiación y el procedimiento de revisión. También se establece la obligación de revisarlos cada 4 años y la posibilidad de articularlos mediante convenios de colaboración entre la

Administración General del Estado y las Comunidades Autónomas (CCAA). Finalmente, se abre la posibilidad a las Entidades Locales de elaborar Planes de Gestión de Residuos Urbanos, de acuerdo con la legislación vigente en materia de competencias municipales, y los planes correspondientes de las respectivas Comunidades Autónomas.

Con el fin de eliminar la relación existente entre crecimiento económico y producción de residuos, la Unión Europea se dota con una directiva de marco jurídico para controlar todo el ciclo de residuos, desde su producción a su eliminación, y se centra, para ello en la valorización y el reciclaje.

La Directiva 2008/98/CE del Parlamento Europeo y del Consejo, de 19 de noviembre de 2008, sobre los residuos y por la que se derogan determinadas Directivas. Esta Directiva establece su marco jurídico para el tratamiento de los residuos en la Unión Europea. Su objetivo es proteger el medio ambiente y la salud humana mediante la prevención de los efectos nocivos que suponen la producción y la gestión de residuos.

Los Estados miembros de la Unión Europea han de tomar las medidas que procedan para fomentar la reutilización de los productos y para promover un reciclado de alta calidad. Para ello establece unos plazos temporales en que los miembros deben de cumplir con estas normativas.

Cada tres años los Estados miembros informarán a la Comisión sobre su situación en lo que se refiere al logro de los objetivos. Si no se cumplen los objetivos, este informe incluirá los motivos de dicho incumplimiento y las medidas que el Estado miembro piensa adoptar para alcanzar dichos objetivos.

De acuerdo con el principio de quien contamina paga, los costes relativos a la gestión de los residuos tendrán que correr a cargo del productor inicial de residuos, del poseedor actual o del anterior poseedor de residuos.

Los Estados miembros podrán decidir que los costes relativos a la gestión de los residuos tengan que ser sufragados parcial o totalmente por el productor del producto del que proceden los residuos y que los distribuidores de dicho producto puedan compartir los costes.

Los primeros planes nacionales de residuos en España se remontan a 1995, fecha en la que se aprobaron el I Plan Nacional de Residuos Peligrosos (1995 – 2000) y el I Plan Nacional de Recuperación de Suelos Contaminados (1995 – 2005). En estos planes se preveían inversiones en la mejora de la gestión, creación de infraestructuras y confección de inventarios.

A lo largo de todos estos años, las Comunidades Autónomas y Ciudades Autonómicas también han ido elaborando y aprobando Planes estratégicos sobre

gestión de residuos, de contenidos y alcances variados, en función de sus propias políticas y prioridades.

En Canarias la Ley 1/1999, de 19 de Enero, de Residuos de Canarias establece que la gestión de residuos tiene como finalidad evitar los perjuicios para los sistemas ambientales, los recursos naturales y el paisaje, erradicar y paliar molestias para las poblaciones, dar un tratamiento ambiental adecuado a las operaciones de eliminación, recuperar suelos contaminados, eliminar los vertederos no autorizados y controlar e integrar los vertederos colmatados.

El primer Plan Integral de Residuos de Canarias fue elaborado por la Consejería de Política Territorial y Medio Ambiente del Gobierno de Canarias, en el periodo 1994 – 1005 y aprobado por Acuerdo del Gobierno de la Comunidad el 13 de Mayo de 1997 (BOC nº 22, de 18 de Febrero de 1998).

Dicho Plan se concibió como el instrumento que debía servir para aplicar a la gestión de residuos en Canarias las directrices emanadas del V Programa de Acción en materia de Medio Ambiente y Desarrollo Sostenible, a la vez que se tienen en cuenta las particularidades propias de todas y cada una de las islas del archipiélago canario.

Pese a estar vigentes los objetivos y las líneas de actuación que conforman el primer Plan Integral de Residuos de Canarias (PIRCAN) (2000 – 2006), aprobado por Decreto 161/2001, de 20 de Julio (BOC nº 134 de 15/10/2011). La aprobación de nueva legislación europea, estatal y autonómica, ha aconsejado realizar una actualización del mismo, al amparo de lo establecido en el Artículo 10 – Tramitación y revisión del Plan, de la Ley de Residuos de Canarias.

Para todos los residuos contemplados en el Plan se pretende una adecuada gestión, que contempla necesariamente las siguientes actividades jerarquizadas:

- Reducción de la producción de residuos, desarrollando o potenciando aquellas actuaciones que minimicen la cantidad de residuos generados, bien por menor empleo de materiales, mediante cambios introducidos en los procesos productivos, o porque permitan un mejor uso de los mismos.
- Utilización de materiales que una vez desechados conduzcan a residuos que tengan menor peligrosidad o que presenten menos problemas para su eliminación.
- Reutilización, reciclado y valorización de los productos contenidos en los distintos tipos de residuos, dentro de los habituales esquemas de gestión para estas actividades.
- Eliminación de la fracción o fracciones de residuos, que no pueden ser evitados o valorizados, de una forma segura para la salud de las personas y el medio ambiente.

En definitiva, el Plan Integral de Residuos de Canarias ha de servir de marco de referencia para instrumentar todas las actuaciones necesarias para una correcta

gestión de los Residuos generados y gestionados en el ámbito de la Comunidad Autónoma de Canarias y de acuerdo con lo recogido en el Plan Nacional de Residuos Urbanos, Plan Nacional de Residuos Especiales y Plan Nacional de Residuos Peligrosos.

La Ley de Residuos de Canarias establece el régimen jurídico de los residuos que se generen o importen en su ámbito territorial, con el fin de garantizar la protección de la salud, la defensa del medio ambiente y la protección de los recursos naturales mediante la ordenación y gestión de los residuos, de acuerdo con las políticas asumidas a nivel estatal y comunitario en esta materia. Además han tenido en cuenta otras disposiciones de carácter legal de Ordenación del Territorio y de Protección Ambiental específicas o no de los residuos, que pueden afectar al presente Plan.

En lo concerniente a la gestión de residuos en las Islas Canarias es preciso tener en cuenta además, una serie de condiciones específicos a la hora de aplicar las directrices europeas en esta materia. Dichos condicionantes inciden de forma determinante sobre todo el proceso de gestión, teniendo como resultado final un encarecimiento, tanto en los costes de inversión como en los gastos de explotación, además de la necesidad de periodos de tiempo más largos para la implantación de los sistemas avanzados de gestión.

El ajuste a la realidad canaria implica la necesidad de considerar los Condicionantes Específicos siguientes:

- Región ultraperiférica de la Unión Europea – Doble Insularidad.
- Importancia del sector turístico en la economía canaria.
- Grado de protección del territorio.

Cada uno de estos condicionantes, por sí solos y en conjunto, dan lugar a una serie de aspectos, factores de incidencia, que afectan directamente a la planificación y gestión de los residuos que se reflejan en el diagrama adjunto y que a continuación se analizan.

Por otra parte, las Islas canarias se consideran región ultraperiférica de la Unión Europea, y tienen un trato de doble insularidad. Frente a la estructura administrativa, el hecho diferencial presenta un escalón más que en el resto de Comunidades Autónomas del Estado Español, al disponer cada una de las islas que conforman el archipiélago de cierta autonomía, tanto en la planificación, como en la gestión de los residuos que se generan en un ámbito geográfico.

De acuerdo con la Ley de Residuos de Canarias cada isla debe disponer de un Plan Director Insular de Residuos, que se integrará posteriormente en un plan autonómico de gestión (Plan Integral de Residuos de Canarias). Ello implica en la práctica procesos más largos en la toma de decisiones.

Además, el diferente desarrollo poblacional alcanzado en cada una de las islas impide el aprovechamiento del "factor de escala" en todo el proceso de gestión de residuos,

recogida, transporte y tratamiento, al menos de las denominadas islas no capitalinas (Lanzarote, Fuerteventura, La Gomera, La Palma y El Hierro). Siendo necesario además el disponer de infraestructuras de almacenamiento y transferencia para cierto tipo de residuos (peligrosos y valorizables) que han de ser transportados fuera del ámbito donde se generan.

Otro factor influyente es la escasez de territorio. Este factor influye de forma decisiva en la dificultad de ubicar instalaciones para la gestión y eliminación de residuos. Además y principalmente en las denominadas islas capitalinas (Gran Canaria y Tenerife) a este factor hay que sumarle las altas densidades demográficas y por tanto la elevada ocupación del suelo, además de otros factores, orográficos (comunicaciones), medioambientales, socioeconómicos (turismo), etc., comunes al conjunto del archipiélago.

Al ser un territorio formado por islas y alejado del continente (Región Ultraperiférica) hay determinados sobrecostes de gestión, tanto en lo referente al transporte interinsular, como del archipiélago al continente, para aquellas fracciones de residuos (peligrosos o reciclables) que requieren de su traslado a península para posibilitar su tratamiento o valorización.

La importancia del sector turístico es otro factor a tener en cuenta. El elevado número de turistas que visitan las islas (en torno a 11 millones anuales sobre una población censada de 1,7 millones) incide de forma decisiva en la generación de residuos y por lo tanto en el dimensionamiento de equipamientos e instalaciones y en gastos de explotación necesarios para una gestión avanzada en esta materia. Por otra parte al tratarse de ciudadanos, principalmente centroeuropeos, con un elevado nivel de concienciación cívica, tienen en cuenta a la hora de elegir destino los aspectos relacionados con la protección del entorno, entre los que se encuentra la gestión de los residuos.

Cabe destacar el grado de protección del territorio. El elevado número de Espacios Protegidos, aproximadamente un 40% de la superficie del archipiélago, restringe la posibilidad de implantación de instalaciones de tratamiento de residuos, además de exigir garantí as medioambientales adicionales, a sumar a la elevada ocupación del suelo, sobre un territorio de por sí ya escaso.

La organización de la recogida de residuos es fundamental para llevar a cabo todas estas leyes. Organizar según la Real Academia Española, significa “Establecer o reformar algo para lograr un fin, coordinando a las personas y los medios adecuados”.

Las empresas de recogida de residuos tienen que construir estrategias optimizadas para la recogida de residuos teniendo en cuenta restricciones como los costes de la recogida. Estos costes no sólo se corresponden a la distancia que recorren dichos vehículos, sino que también se tiene en cuenta el consumo de los vehículos de la flota, la orografía del terreno, y los costes de personal. Además, la recogida de

residuos tiene que tener en cuenta el tiempo que tarda en realizar todas las rutas para la recogida, ya que debe de ser mínimo, y el horario en que se realiza dicha recogida. Es importante tener en cuenta el horario ya que si los vehículos recorrieran sus rutas a lo largo del día, podrían encontrarse con dificultades en el tráfico que les retrasarían, pudiendo incluso llegar a impedirles recorrer algunas de las calles. Así pues, dado que es más conveniente realizar esta recogida en horario nocturno, es importante volver a incidir en reducir el tiempo que tardan los vehículos, ya que podrían llegar a molestar a los ciudadanos.

Cada empresa de recogida decide qué periodo de tiempo es conveniente para organizar las rutas. Esta organización puede venir determinada por el volumen de residuos que genera la población, y por cómo organicen la recogida de residuos selectivos. Esta recogida de residuos selectivos puede ser:

Contenedores de superficie

Este sistema es el que se desarrolla de forma más habitual. Consiste en ubicar en la vía pública contenedores de diferente tipología, dependiendo de las características de la fracción a recoger y del urbanismo de la zona. Para aportar los residuos separados en origen, los ciudadanos deben desplazarse a los puntos de recogida más o menos cercanos y agrupados por fracciones. Estos contenedores son vaciados por los correspondientes servicios de recogida siguiendo unos horarios y frecuencias adaptadas a los niveles de llenado de los receptáculos, según la generación de cada fracción y en algunos casos también en función de las características del propio material (olores en el caso de la fracción orgánica de recogida separada (FORS).

Contenedores soterrados

La recogida separada en contenedores soterrados consiste en ubicar los contenedores bajo el nivel del suelo de manera que únicamente queda en superficie el buzón a través del cual se depositan los residuos y la tapa que se debe abrir para elevar el receptáculo interno. La instalación de los contenedores soterrados supone realizar obra civil en la vía pública. Existen numerosos modelos de contenedores soterrados que se diferencian básicamente por la tipología de receptáculo utilizado, el diseño de los buzones y por el sistema de elevación (elevación por pluma o sistema hidráulico para elevar la tapa y/o el receptáculo).

Normalmente los contenedores son de gran volumen (habitualmente de 3.000L o 5.000L, aunque también dependiendo del sistema de elevación, se pueden incorporar contenedores de ruedas en su interior equivalentes a los de superficie. Adicionalmente, existe un sistema de contenedores semi-soterrados para los que parte del receptáculo se encuentra bajo el nivel del suelo y por tanto permite también disponer de grandes capacidades.

Puerta a Puerta

La recogida separada puerta a puerta (PaP) consiste en entregar los residuos al servicio municipal de recogida delante de la puerta de la vivienda o comercio (en bolsas, pequeños contenedores-normalmente para la FORS- o a granel- para el papel y cartón en cajas o fardos), según un calendario semanal para cada fracción recogida y en un horario estipulado.

Mediante este sistema se pueden recoger todas las fracciones tanto domésticas como comerciales. El sistema, como mínimo, deber recoger puerta a puerta el Resto y la FORS domiciliaria. Los residuos de envases de vidrio domiciliario es la única fracción que se suele mantener en sistema de contenedores. Este sistema PaP es muy utilizado para las recogidas específicas comerciales ya que se consigue una gran calidad y cantidad del material y evitando problemas de desbordamiento o necesidad de gran capacidad de recepción de los contenedores de vía pública domiciliarios.

Los resultados de recogida separada logrados en los municipios con PaP son en general superiores al resto de mecanismos, tanto en cantidad recogida como en calidad de la separación (en general se sitúan entre el 60 y el 80% de recogida separada). La aplicación del PaP es muy recomendada en zonas de baja densidad de población donde la identificación de los residuos de cada cual es más fácil, pero también se puede desarrollar en zonas más densas y con edificación más vertical a través de la recogida de contenedores comunitarios en las viviendas plurifamiliares. Los modelos de recogida PaP permiten identificar al generador y por lo tanto posibilitan la implantación de sistemas de fiscalización más justos cómo son los de pago por generación (por ejemplo, pago por bolsa o pago por contenedor).

Neumática

El sistema de recogida neumática de residuos consiste en una serie de buzones de vertido conectados mediante tuberías subterráneas al punto de captura desde donde se realiza una aspiración del circuito.

Los buzones de recogida se pueden ubicar tanto en el interior de las viviendas, en áreas comunitarias dentro de los edificios, como en áreas públicas exteriores en acera o en la misma fachada de los edificios. Los residuos vertidos en los buzones caen por gravedad hasta las válvulas que están instaladas en niveles inferiores, y allí se acumulan temporalmente hasta que se realiza el proceso de aspiración.

Por otro lado, según el caso, las diferentes fracciones a recoger se pueden depositar en el mismo buzón si se utiliza un sistema diferenciado de bolsas de diferentes colores que posteriormente se separan en la central, o bien, en buzones diferentes que se aspiran de forma independiente.

Puntos limpios

Los Puntos Limpios (en algunas zonas llamados Ecocentros, Ecoparques, Puntos Verdes, etc.) son centros de aportación y almacenamiento, selectivos, principalmente de residuos de competencia municipal que no son objeto de recogida domiciliaria y tienen el objetivo de facilitar la gestión correcta de las fracciones no ordinarias. Estas instalaciones son principalmente para uso de particulares y pequeños comercios (incluso de pequeños industriales y servicios municipales) de acuerdo con las especificaciones de las correspondientes ordenanzas municipales.

Con el fin de construir estas estrategias optimizadas para la recogida de residuos teniendo en cuenta las restricciones detalladas, las empresas tienen a su disposición herramientas software disponibles para la optimización de rutas de vehículos, como por ejemplo la herramienta 4gflota. A través de esta, las empresas introducen todas las restricciones de su problema (el número de vehículos disponibles, la capacidad de carga de cada uno, o los clientes a los que hay que atender junto con su ubicación geográfica) y obtienen como resultado unas rutas optimizadas que se ajustan a su problema concreto.

La optimización de la logística de recogida es imprescindible para obtener unos buenos resultados de gestión, ambientales y económicos. Las tipologías de servicio (recogidas puerta a puerta, comerciales o a demanda, etc.), la dispersión territorial y las características de algunas fracciones (por ejemplo de la FORS o los residuos de envases ligeros), pueden hacer que el consumo en la etapa de recogida sea importante (y las emisiones derivadas también). Los siguientes mecanismos pueden optimizar el funcionamiento del servicio:

- Utilización de vehículos biocompartimentados.
- Desarrollo de circuitos compartidos con las recogidas comerciales.
- Implantación de la recogida de la FORS de forma integrada (no aditiva) al conjunto de recogidas.
- Optimización de rutas y frecuencias en función de las necesidades de cada zona.
- Utilización de vehículos con capacidades adecuadas a las toneladas recogidas y si es posible compactadores.
- Uso de vehículos más eficientes y combustibles menos contaminantes (biodiesel, gas-natural, vehículos híbridos, eléctricos-actualmente disponibles esencialmente para vehículos de pequeña capacidad).
- Implantación de sistemas de transferencias en plantas de transvase locales o mancomunadas.

Finalmente, resulta interesante calcular el balance energético y de emisiones de los circuitos de recogida hasta las plantas para detectar el impacto y viabilidad de la recogida y optimizar los servicios.

CAPÍTULO 3

3. PROBLEMAS DE RUTAS DE VEHÍCULOS

Desde este capítulo se busca ilustrar acerca de los diferentes modelos existentes que estudian los problemas de rutas de vehículos, exponiendo la relación entre cada uno de ellos. Se incluye también la formulación matemática de cada uno de ellos.

Muchos problemas de optimización consisten en obtener un conjunto de valores o configuración de variables del problema que satisfagan condiciones intrínsecas del problema y a la vez cumplan con determinados objetivos. Los problemas que cumplen estas condiciones se agrupan en dos clases: aquellos cuyas soluciones deben codificarse con variables continuas y aquellas cuyas soluciones deben codificarse con variables discretas. Los primeros son problemas de optimización continua y los segundos, problemas de optimización combinatoria.

Los problemas de optimización combinatoria parten de disciplinas que hacen uso de métodos numéricos como la inteligencia artificial, la bioinformática y el comercio electrónico. Los problemas más conocidos se relacionan con la logística, el diseño de hardware, el secuenciamiento de cadenas de ADN, la planificación horaria,… En estos problemas, en general se deben encontrar agrupamientos, ordenamientos o asignaciones óptimas de un conjunto discreto y finito de objetos que satisfagan determinadas condiciones o restricciones.

Una forma de resolver los problemas de optimización combinatoria es dada una instancia del problema, buscar todas las soluciones en el espacio de soluciones. Esto es, enumerar todas las soluciones posibles y luego elegir la mejor. A pesar de que esta forma de resolver parece lógica y simple, en la mayoría de los casos, esta propuesta se vuelve rápidamente impracticable porque el número de soluciones crece de manera exponencial según el tamaño de la instancia. En algunos problemas de optimización combinatoria, el estudio y análisis de la estructura del problema y las características de las instancias del problema, permiten desarrollar algoritmos que alcanzan soluciones óptimas en menor tiempo computacional que la búsqueda exhaustiva. En otros problemas, estos mismos algoritmos no logran un rendimiento mucho mejor que la búsqueda exhaustiva.

Dada la importancia práctica de los problemas de optimización combinatoria, se han desarrollado diferentes algoritmos con el objetivo de obtener soluciones a los mismos. Estos algoritmos se clasifican en algoritmos exactos y algoritmos heurísticos.

Los algoritmos exactos garantizan obtener la solución óptima a un problema de optimización en un tiempo computacional dependiente de la instancia del problema, siendo la misma de tamaño finito. Sin embargo, para muchos problemas de tipo NP-Hard, los algoritmos exactos necesitan un tiempo exponencial para alcanzar las soluciones óptimas. Incluso para instancias pequeñas, estos algoritmos podrían consumir demasiado tiempo computacional, lo cual puede resultar poco práctico.

Los algoritmos heurísticos obtienen soluciones cercanas a la solución óptima, pero a bajo costo computacional. Una clasificación básica es dividirlos en algoritmos constructivos y algoritmos de búsqueda local. Los algoritmos constructivos generan soluciones a un problema agregando elementos de forma iterativa a una solución parcial, hasta que la misma es completada. En cambio, los algoritmos de búsqueda local examinan el vecindario de una solución inicial con el objetivo de alcanzar el óptimo local.

3.1 PROBLEMA DEL VIAJANTE DE COMERCIO

El Problema del Viajante de Comercio (Travelling Salesman Problem o TSP) surge en 1930 dentro de los estudios sobre problemas de rutas. Es uno de los problemas de optimización más importantes dentro del campo de la investigación operativa, y a partir de este aparecen muchos estudios sobre problemas similares y de gran interés.

Dado un conjunto de ciudades y conocida la distancia para viajar entre ella, el objetivo del TSP es encontrar la ruta de menor coste que, partiendo y acabando en una ciudad concreta, visite exactamente una vez cada una de ellas.

El TSP puede ser modelado como un grafo no dirigido ponderado (TSP simétrico), de tal manera que las ciudades son los vértices del grafo y el coste del viaje son las longitudes de las aristas. Así, en Teoría de Grafos el problema consiste en encontrar un ciclo Hamiltoniano. Aunque por lo general se estudia el TSP simétrico, también es posible considerar el TSP asimétrico, en donde la distancia de ir de una ciudad a otra no es la misma que en el sentido contrario. Esto es posible ya que en la realidad existen calles de un único sentido, calles cortadas por obra,… por lo tanto el problema simétrico resulta insuficiente.

El TSP tiene un planteamiento muy sencillo y aunque es muy difícil de resolver, existen muchas heurísticas y métodos exactos con los que se consigue resolver para un gran número de ciudades. La formulación más popular del TSP es la dada por

Dantzing, Fulkerson y Johnson [3], que resolvieron el problema para 49 ciudades. Además de estos, existen muchos otros investigadores que han intentado aportar herramientas para resolver ejemplos mucho mayores (hasta el momento se han resuelto ejemplos de hasta 24978 ciudades).

Sea un grafo no dirigido $G = (E, V)$ donde V es el conjunto de ciudades con $|V| = n$ y E es el conjunto de aristas entre cada dos ciudades. Para cada $e \in E, e = [i, j], \forall\, i, j \in V$, donde i es la ciudad de partida y j la ciudad de destino. Así, el modelo dado en [1] es formulado como un modelo de programación lineal entera binario, donde cada arista $e = [i, j] \in E$ tiene asociadas una distancia c_e (distancia entre dos ciudades) y la siguiente variable:

$$x_e = \begin{cases} 1\ si\ e\ pertenece\ a\ la\ ruta \\ 0\ en\ otro\ caso \end{cases} \quad \forall\, e \in E$$

De esta forma, el modelo es formulado como sigue:

$$mín \sum_{i,j \in V} c_{[i,j]} x_{[i,j]} \qquad (3.1)$$

Sujeto a

$$\sum_{j \in V \setminus \{i\}} x_{[i,j]} = 2 \qquad \forall i \in V \qquad (3.2)$$

$$\sum_{i \in S, j \notin S} x_{[i,j]} = 2 \qquad (3.3)$$

$$0 \leq x_{[i.j]} \leq 1 \qquad \forall i, j \in V \qquad (3.4)$$

$$x_{[i,j]}\ entera\ \forall i, j \in V \qquad (3.5)$$

Las restricciones (3.2) establecen que cada ciudad debe ser visitada una única vez y las restricciones (3.3) son restricciones de eliminación de subtour. Estas últimas restricciones hacen que sea prácticamente imposible resolver directamente el problema (3.1)-(3.5). Aunque es posible encontrar soluciones mediante el método de ramificación y corte (“branch and cut"), resolviendo al principio el problema sin las restricciones (3.3) y luego añadiéndolas si realmente son necesarias.

El Problema del Viajante de Comercio es uno de los problemas de Optimización combinatoria más estudiados. De manera general, dado un grafo ponderado, conexo y dirigido, TSP consiste en encontrar el camino cíclico más corto tal que todo nodo del grafo sea visitado exactamente una vez. A los efectos de definir formalmente TSP, estas son unas definiciones previas:

Camino y ciclo Hamiltoniano

Sea $G = (V, E, w)$ un grafo ponderado, conexo y dirigido donde $V = \{v1, v2, \dots, vn\}$ es el conjunto de $n = |V|$ vértices, $E \subseteq V\ x\ V$ el conjunto de aristas (arcos), y una función $w: E \to \Re^+$ que asigna toda arista $e \in E$ un peso $w(e)$.

Un camino cíclico en G es un camino tal que el primer y el último vértice coinciden.

Un ciclo hamiltoniano (o circuito) es un camino cíclico en G que visita todos los vértices del grafo exactamente una vez (excepto el vértice inicial); es decir, $p = (u1, u2, \dots, uk)$ es un ciclo hamiltoniano en G sii $n = |V|$ y $\{u1, u2, \dots, un\} = V$.

Peso (costo) de un camino

Dado un grafo G ponderado, conexo y dirigido, y un camino $p = (u1, u2, \dots, uk)$ en G, el peso de un camino $w(p)$ se define como $w(p) = \sum_{i=1}^{k-1} w(\left(u_{[i]}, u_{[i+1]}\right))$.

Puede definirse entonces: Dado un grafo G ponderado, conexo y dirigido, el TSP consiste en encontrar un ciclo hamiltoniano en G con peso mínimo.

El interés en este problema de tipo NP-Hard se debe entre otros, a que es un problema conceptualmente simple de explicar y entender, con aplicaciones reales concretas, generalmente relacionadas con problemas de rutas de vehículos, y continuamente se proponen diferentes variaciones al problema (Múltiple TSP, CVRP, VRPTW,…).

3.2 Problema de Rutas de Vehículos

El Problema de Rutas de Vehículos (Vehicle Routing Problem o VRP) es una generalización natural del TSP, en donde los clientes demandan una cantidad de productos que deben ser entregados con una flota de vehículos, y cada vehículo tiene su propia ruta. Este problema es un poco más difícil que el TSP pues es necesario particionar el conjunto de clientes para que sean atendidos separadamente por cada vehículo y después determinar el orden de servicio de cada vehículo.

El VRP es otro ejemplo de problemas de Optimización combinatoria. Fue propuesto por Dantzig y Ramser [4] en 1959, quienes describieron una aplicación real sobre el suministro de gasolina en estaciones de servicio y propusieron la primera formulación matemática y un método algorítmico para su resolución. El VRP es una generalización natural del TSP, en donde los clientes demandan una cantidad de productos que deben ser entregados con una flota de vehículos. Debido a esto, el VRP también es un problema HP-Hard y por ello, y por la gran cantidad de

aplicaciones prácticas que existen, a partir del VRP han surgido muchos modelos y algoritmos para resolver diferentes versiones de él.

El VRP con capacidades es la versión más simple y más estudiada del VRP. Este consiste en minimizar el coste total de todas las rutas que hace una flota de vehículos para servir a un conjunto de clientes, debiendo partir del depósito y donde la capacidad del vehículo está limitada. El VRP con capacidades también puede ser formulado como un problema de Teoría de Grafos. Sea $G = (E, V)$ es un grafo no dirigido, donde $V = \{0, 1, \dots, n\}$ es el conjunto de vértices, siendo 0 el depósito de vehículos, y E es el conjunto de aristas entre cada dos vértices. Para una arista $e = [i, j] \in E$ denotamos por c_e la distancia de ir de i a j. Sea también una flota de k vehículos, cada uno de capacidad Q y d_i la demanda del cliente i. Además, se define una variable de la siguiente manera:

$$x_e = \begin{cases} 1 \; si \; e \; pertenece \; a \; la \; ruta \\ 0 \; en \; otro \; caso \end{cases} \quad \forall \, e \in E$$

De esta forma, el modelo es formulado como sigue:

$$mín \sum_{i,j \in V} c_{[i,j]} x_{[i,j]} \qquad (3.6)$$

Sujeto a

$$\sum_{j \in V \setminus \{i\}} x_{[i,j]} = 2 \qquad \forall i \in V \setminus \{0\} \qquad (3.7)$$

$$\sum_{j \in V \setminus \{0\}} x_{[0,j]} = 2k \qquad (3.8)$$

$$\sum_{i \in S, j \notin S} x_{[i,j]} \geq \frac{2}{Q} \sum_{i \in S} d_i \quad \forall i \in V \setminus \{0\}, \qquad (3.9)$$

$$0 \leq x_{[i.j]} \leq 1 \qquad \forall i, j \in V$$

$$x_{[i,j]} \; entera \qquad \forall i, j \in V$$

Las restricciones (3.7) establecen que cada ciudad debe ser visitada una única vez y las restricciones (3.8) establecen que cada vehículo sale una única vez del depósito y entra una única vez en el depósito. Las restricciones (3.9) son restricciones de eliminación de subtour y además, establecen que un conjunto de clientes que supera la capacidad máxima *Q* no pueda ser visitado por el mismo vehículo. Las soluciones de este problema son todas las *k*-rutas posibles que verifican la restricción de capacidad de los vehículos.

Todos estos problemas tienen numerosas aplicaciones prácticas. A parte de las más evidentes en el área de la logística de trasporte, existen otros ejemplos como en control y operativa de semáforos, en robótica, que permite resolver problemas de fabricación, etc.

A pesar de que existen muchas investigaciones en esta área, los problemas de rutas más básicos son todavía muy difíciles de resolver óptimamente, incluso para tamaños relativamente pequeños y de escasa utilidad práctica. Así, muchos estudios están centrados en el diseño de algoritmos para su resolución.

Los problemas de rutas de vehículos en realidad son un amplio conjunto de variantes y personalizaciones de problemas. En ellos se trata de averiguar las rutas de una flota de transporte para dar servicio a unos clientes. Este tipo de problemas pertenece a los problemas de optimización combinatoria.

La función objetivo depende de la tipología y características del problema. Lo más habitual es intentar: minimizar el coste total de operación, minimizar el tiempo total de transporte, minimizar la distancia total recorrida, minimizar el tiempo de espera, maximizar el beneficio, maximizar el servicio al cliente, minimizar la utilización de vehículos, equilibrar la utilización de recursos,…

Hay numerosas variantes del Vehicle Routing Problem, tales como el VRP Capacitado (CVRP), VRP Capacitado con ventanas de tiempo (CVRPTW), VRP con entregas y recogidas (VRPPD), VRP Periódico (PVRP), VRP Periódico con ventanas de tiempo (PVRPTW),…

3.3 Problema del Cartero Chino

Históricamente, el estudio sobre el problema de rutas de arcos comienza con la presentación de la solución al problema de Königsberg en 1735 por parte de Euler. El problema es el siguiente: sea un grafo conexo $G = (V, E)$, encontrar un recorrido que visite cada arista de E exactamente una vez o bien informar que no es posible obtenerlo. Euler probó que es posible obtener tal recorrido siempre que el grado de cada nodo de V sea par.

El siguiente problema es el Problema del Cartero Chino (Chinese Postman Problem o CPP) el cual fue inicialmente propuesto por Kwan Mei-Ko en 1962. Se define así: sea un grafo conexo $G = (V, E, w)$ donde w es una matriz de costos, encontrar un recorrido tal que visite cada arista al menos una vez al menor costo posible. Edmonds y Johnson probaron que si el grafo asociado al problema es dirigido o no dirigido, CPP puede ser resuelto en tiempo polinomial. Sino es un problema NP-Hard [5][6].

En 1974, Orloff propuso el Problema del Cartero Rural (Rural Postman Problem o RPP) el cual se define de la siguiente manera: sea un grafo no dirigido $G = (V, E, w)$ donde w es una matriz de costos, encontrar un recorrido de costo mínimo tal que éste visite cada arista de $R \subseteq E$ al menos una vez.

La diferencia entre CPP y RPP es que en el problema del Cartero Rural sólo un conjunto de aristas deben ser visitadas. En 1976, Lenstra y Rinnoy Kan demostraron que RPP es NP-Hard.

A continuación se define el Problema de Rutas de Arcos Capacitados (CARP) el cual es una extensión del problema del Cartero Rural, con restricciones de capacidad.

3.4 Problema de Rutas de Arcos Capacitados

El Problema de Rutas de Arcos Capacitados (Capacitated Arc Routing Problem o CARP) consiste en atender las demandas sobre determinadas calles de una red vial a través de una flota homogénea de vehículos, los cuales inician y finalizan sus recorridos desde un único depósito. El objetivo del problema es minimizar el costo total del recorrido de forma que se atiendan todas las demandas sin exceder la capacidad de carga de los vehículos involucrados.

Uno de los beneficios del estudio de este problema radica en su aplicación real en organismos públicos y empresas. En ambos casos, se busca ofrecer un servicio de mejor calidad y al menor costo posible. No obstante, es probable que sus problemas de rutas posean características adicionales al CARP, es decir, características relacionadas a su contexto que deben ser consideradas y por tanto, agregadas al modelo original de manera tal que reflejar mejor su realidad. Por ejemplo, minimizar el número de vehículos usados, cumplir con las demandas dentro de un cierto intervalo de tiempo especificado, considerar una red vial con calles en un único sentido, en dos o ambos, utilizar varios depósitos desde donde los vehículos puedan comenzar y finalizar sus recorridos, considerar vehículos con diferente capacidad de carga,…

Es por ello que el estudio del Problema de Rutas de Arcos Capacitados como así también de sus extensiones, está motivado por la necesidad de ofrecer mejores y variadas alternativas de soluciones a aquellas organizaciones que deben tratar cotidianamente con problemas de rutas y logística.

Modelo matemático como problema de programación lineal para CARP [8]. Sea $G = (V, E)$ un grafo, donde V es el conjunto de vértices, E el conjunto de aristas. $R \subseteq E$ es el conjunto de aristas requeridas, $V_r \subseteq V$ es el conjunto de vértices conteniendo los extremos de las aristas de R, más el vértice asociado al depósito. Sea también $K = \{1, \dots, M\}$ es la flota de vehículos con capacidad de carga Q, c_{ij} es el costo de

recorrido de la arista (i,j) perteneciente a E, y q_{ij} es la demanda asociada a la arista (i,j) perteneciente a E. Se define una variable de la siguiente manera:

$$q_{ij} = \begin{cases} 0 & si\ (i,j) \notin R \\ > 0 & si\ (i,j) \in R \end{cases}$$

Las variables de decisión se definen

$$x_{[i,j,k]} = \begin{cases} 1 & si\ (i,j)\ es\ recorrida\ por\ el\ vehículo\ k \\ 0 & en\ otro\ caso \end{cases}$$

El modelo matemático para CARP es el siguiente:

$$Minimizar \sum_{(i,j)\in E} c_{[i,j]} \sum_{k\in K} x_{[i,j,k]} \qquad (3.10)$$

$$\sum_{k\in K} x_{[i,j,k]} = 1 \qquad (i,j) \in R \qquad (3.11)$$

$$\sum_{j\in V_r - \{0\}} \sum_{k\in K} x_{[0,j,k]} = |K| \qquad (3.12)$$

$$\sum_{(i,j)\in R} q_{[i,j]} x_{[i,j,k]} \leq Q \qquad k \in K \qquad (3.13)$$

$$\sum_{j\in V_r} x_{[i,j,k]} = \sum_{j\in V_r} x_{[j,i,k]} \qquad i \in V_r, k \in K \qquad (3.14)$$

$$\sum_{i\in S} \sum_{j\notin S} x_{[i,j,k]} \geq \sum_{j\in V} x_{[h,j,k]} \qquad S \subseteq V_r - \{0\}, k \in K, h \in S \qquad (3.15)$$

$$x_{[i,j,k]} \in \{0,1\} \qquad (i,j) \in R, k \in K$$

Donde la función objetivo (3.10) busca minimizar el costo total del recorrido, sujeta a las restricciones de:

- Atender las demandas de las aristas requeridas por un único vehículo (3.11)
- El número de vehículos usados es el número de vehículos disponibles (3.12)
- Sin exceder la capacidad de carga de ninguno de ellos (3.13)

Además de estas restricciones, se agregan restricciones de igualdad (3.14) y de eliminación de sub-recorridos (3.15).

CARP consiste entonces, en determinar un conjunto de rutas para los vehículos a menor costo total de recorrido, tal que cada ruta comienza y finaliza en el depósito, cada una de las demandas asociadas a las aristas de R son atendidas por un único vehículo y ningún vehículo involucrado excede su capacidad de carga Q.

De acuerdo a la formulación antes descrita, puede observarse cierta relación con el Problema de Rutas de Vehículos. La diferencia reside en que VRP es un problema de rutas sobre nodos o vértices. En la bibliografía se han puesto transformaciones de CARP a VRP [8] y de VRP a CARP (por ejemplo [9]).

Golden y Wong [9] demostraron que CARP es NP-Hard y que una solución dentro de un factor 3/2 del resultado óptimo es NP-Hard. Las propuestas existentes en la literatura para ofrecer una solución a CARP están clasificadas en algoritmos heurísticos, metaheurísticos y algoritmos exactos [10][11][12][13].

Un ejemplo de aplicación del CARP para la recogida de residuos sería el estudio de Ghiani et. al. [14]. Este estudio trata sobre la recolección de residuos domiciliarios para la ciudad de Castrovillari, ubicada al sur de Italia. El problema es una variación de CARP que incluye instalaciones intermedias (sitios de descarga de residuos), ventanas de tiempos para cumplir con el servicio sobre arcos/aristas requeridas y flota de vehículos heterogénea. Los objetivos del estudio se enfocaron en la reducción de rutas, el número de vehículos utilizados y las horas extras del personal a cargo del servicio de recolección.

Se propuso un algoritmo Agrupación-Rutas (Cluster-First Route-Second), el cual en su primera fase asigna arcos y aristas requeridas a vehículos siempre que se cumplan las condiciones de ventanas temporales. En la siguiente, se determinan los recorridos que realiza cada vehículo considerando restricciones de capacidad vehicular y longitud de recorridos del 10.6 %, una reducción de tiempo de trabajo del 13.5 %, la eliminación de las horas extras y una mejor distribución de cargas entre los vehículos. En términos generales, se obtuvo una reducción del 8 % en términos de costos totales.

CAPÍTULO 4

4. METODOLOGÍA PARA RESOLVER EL PROBLEMA

A través de este cuarto capítulo se realiza un estudio de varios de los métodos existentes, haciendo mayor hincapié en los que finalmente se ha decidido utilizar.

Una técnica sencilla y lógica que puede usarse para resolver un problema de optimización, consiste en examinar todas las soluciones factibles del problema, evaluarlas usando la función objetivo y luego elegir la mejor. Esta técnica llamada de enumeración completa o fuerza bruta, a pesar de que es aplicada en diferentes problemas computacionales, puede resultar impracticable incluso en instancias de tamaño medio debido a la gran cantidad de soluciones posibles.

Dentro de la clasificación de algoritmos exactos, Ramificación y Poda (Branch and Bound) es una técnica eficiente de enumeración completa de soluciones, la cual constituye un árbol de soluciones del problema y que, para evitar una búsqueda exhaustiva en el mismo para obtener la solución óptima, reduce esta tarea a través de podas. Esta técnica como otras tales como Ramificación y Coste (Branch and Price) y Ramificación y Corte (Branch and Cut), han sido aplicadas exitosamente a diferentes problemas de optimización combinatoria. Sin embargo, a medida que aumenta el tamaño de las instancias de los problemas, el tiempo computacional para alcanzar soluciones óptimas crece en forma exponencial.

Dada la dificultad subyacente en los problemas de optimización de tipo NP-Hard, la investigación en técnicas heurísticas ha cobrado relevancia en los últimos años. Una heurística es una técnica, consistente en una regla o un conjunto de reglas, que busca y posiblemente encuentra buenas soluciones, a un costo computacional razonable. En cierto sentido, una heurística es aproximada ya que alcanza buenas soluciones a bajo es fuerzo computacional, pero no asegura la optimalidad. Las heurísticas son reglas, criterios, métodos o principios para decidir entre diferentes cursos de acción, cuál es el más efectivo a los efectos de cumplir con un objetivo, como por ejemplo, lograr una solución factible para un problema o mejorar una solución.

A pesar de que los algoritmos heurísticos no aseguren la obtención de soluciones óptimas, puedan no precisar la brecha entre las soluciones devueltas y la óptima o

incluso no aseguren la factibilidad de las mismas, éstas puede alcanzarse en un tiempo razonable y, al igual que la calidad, que las obtenidas por métodos exactos.

Los algoritmos heurísticos se clasifican en algoritmos constructivos y algoritmos de búsqueda local.

Un algoritmo constructivo construye una solución a un problema de Optimización combinatoria de manera incremental. Paso a paso, agrega componentes (o elementos) a la solución parcial hasta obtener una completa. La forma en que los elementos son agregados a la solución en construcción puede ser al azar o bien mediante la aplicación de alguna regla.

Por ejemplo, un algoritmo constructivo para el TSP, construye una solución agregando nodos del grafo de manera iterativa hasta que todos estén incluidos en la misma. Dicha inserción puede ser de forma aleatoria (como propone Inserción Aleatoria o Random Insertion), la que menos incrementa el costo del recorrido total (Inserción más Económica o Cheapest Insertion), aquella cuya distancia a algún nodo en la solución sea máxima (Inserción más Distante o Farthest Insertion), entre otras alternativas posibles.

Un algoritmo de búsqueda local se basa en la exploración iterativa del vecindario de soluciones, para de esta manera mejorar la calidad de la solución actual mediante cambios locales definidos por una estructura de vecindario.

Los algoritmos de búsqueda local iterativamente modifican soluciones (obteniendo de esta manera soluciones vecinas) a un problema de optimización, mediante cambios (es decir, mediante movimientos de elementos de la solución). Una estructura de vecindario define implícitamente los posibles cambios a aplicar a una solución. Los cambios son previamente evaluados usando alguna función de optimización, para de esta manera conducir el proceso de búsqueda.

Las metaheurísticas son un proceso iterativo maestro que guía y modifica la operación de heurísticas subordinadas, para producir soluciones de alta calidad a cualquier problema de optimización.

Existen diferentes características que poseen las metaheurísticas:

- Son estrategias que con pocos cambios, son aplicables a una gran variedad de problemas de optimización.
- Para conducir la búsqueda a soluciones de alta calidad, hacen uso de información específica del problema y/o aprenden a lo largo de su ejecución.
- Pueden subordinar heurísticas constructivas y/o de búsqueda local.
- Incorporan mecanismos para evitar estancamiento en óptimos locales.

En base a las características de cada metaheurística, es posible obtener diferentes clasificaciones:

Origen

Según la fuente de inspiración de la metaheurística, se clasifican en inspiradas en la naturaleza y no inspiradas en la naturaleza. En el primer grupo se encuentran los Algoritmos Genéticos, Optimización de Colonias de Hormigas (Ant Colony Optimization), Optimización del Apareamiento de las Abejas (Honey-bee Mating Optimization),… En el segundo podemos mencionar a Búsqueda Tabú (Tabu Search), Búsqueda Local Iterativa (Iterated Local Search), Búsqueda Local Guiada (Guided Local Search),…

Manejo de soluciones

Según el número de soluciones que manejan al mismo tiempo durante el proceso de búsqueda, se clasifican en basadas en poblaciones como por ejemplo Algoritmos Genéticos, Optimización del Apareamiento de las Abejas (Honey-bee Mating Optimization) y Algoritmos Genéticos de Clave Aleatoria (Biased Random Key Genetic Algorithm), y las basadas en búsqueda simple como la Búsqueda Local Iterativa (Iterated Local Search), Búsqueda de Vecindario Variable (Variable Neighborhood Search) y Búsqueda Tabu (Tabu Search).

Uso de memoria

Otra forma de clasificar metaheurísticas se relaciona con el uso o no de memoria para registrar el proceso de búsqueda. Algunas metaheurísticas que no incorporan mecanismos de memoria son Variable Neighborhod Search, Simulated Annealing y GRASP. En cambio, Búsqueda Tabu (Tabu Search) y Optimización de Colonias de Hormigas (Ant Colony Optimization) incorporan mecanismos de memoria de corto y largo plazo.

Estructura de vecindario

En general, las metaheurísticas usan una única estructura de vecindario (Búsqueda Local Iterativa (Iterated Local Search), y Optimización de Colonias de Hormigas (Ant Colony Optimization),…). En cambio, Búsqueda de Vecindario Variable (Variable Neighborhood Search) usa diferentes estructuras de vecindarios permitiendo así diversificar la búsqueda de soluciones y evitar estancamiento en óptimos locales mediante mecanismos eficientes.

Movimientos

La forma en que construyen una solución o que exploran el vecindario de una solución puede ser de manera aleatoria o determinística. Optimización de Colonias de Hormigas (Ant Colony Optimization) y GRASP construyen soluciones aplicando reglas probabilísticas. Búsqueda Local Iterativa (Iterated Local Search) elige un vecino de una solución en forma aleatoria. En tanto, Búsqueda Tabu (Tabu Search) en su versión simple construye soluciones usando una memoria de corto plazo que evita incluir elementos que fueron recientemente utilizados.

4.1 Procedimiento de Búsqueda Voraz Adaptativo Probabilista

Una de las metaheurísticas que se utilizan es el Procedimiento de Búsqueda Voraz Adaptativo Probabilista (Greedy Randomized Adaptative Search Procedure o GRASP) encuentra soluciones aproximadas (de buena calidad, no óptimas) a problemas de optimización combinatoria. Tiene como premisa la generación de soluciones iniciales diversas y de calidad, que le permitan a los algoritmos de búsqueda local alcanzar mejores óptimos locales. Esta será la metaheurística propuesta para resolver el problema de este proyecto.

GRASP es un proceso iterativo. Cada iteración consiste en dos fases. Una fase constructiva, en la cual una solución es construida, y otra fase de búsqueda local, en la cual un óptimo local en el vecindario de la solución construida previamente es obtenido. Una vez cumplida la condición de parada, el algoritmo retoma la mejor solución alcanzada.

El principio a cumplir en los algoritmos basados en GRASP es la generación de muchas soluciones iniciales de buena calidad a través de una fase constructiva de soluciones, para luego mediante una fase de mejora, aumentar la probabilidad de encontrar óptimos locales de calidad. GRASP se ha aplicado a diferentes problemas de optimización tales como problemas de agrupamiento, planificación de máquinas, asignación cuadrática, árbol mínimo capacitado, de corte, rutas de vehículos,…

Algoritmo GRASP
mientas (el criterio no satisfecho)hacer *Construye una solución inicial usando el procedimiento voraz* *Realiza una búsqueda local para mejorar la solución construida* *fin mientras*

Figura 2: Pseudocódigo del algoritmo GRASP.

Fase constructiva

En la fase de construcción de soluciones, cada solución es iterativamente construida, agregando un elemento candidato a la vez. Un elemento candidato es aquel que puede ser elegido para formar parte de la solución parcial en construcción. Para determinar cuál elemento candidato debe seleccionarse, se usa una función golosa (greedy o miope) la cual mide la contribución que hace el elemento a la solución parcial.

Para evitar que la selección sea miope y determinística, el elemento a incorporar a la solución se elige de forma aleatoria. Sin embargo, esta selección aleatoria no se realiza con todos los elementos candidatos posibles, sino con los mejores. La Lista Restringida de Candidatos (Restricted Candidate List o RCL), es una lista con buenos elementos candidatos según sus valores obtenidos con la función miope usada. Entonces, el elemento a agregar a la solución es elegido al azar desde esta lista, siendo la misma de tamaño fijo o de tamaño variable dependiendo de los elementos cuyos valores de función miope se encuentren dentro de un intervalo o umbral previamente determinado. Esta técnica permite que se obtengan diferentes soluciones en cada iteración de GRASP, la cual trabajando con funciones golosas apropiadas, permiten que las soluciones también sean de calidad. En general, este proceso se repite hasta obtener una solución al problema. En caso de que esto no ocurra, debe aplicarse algún operador de reparación sobre la solución parcial.

Algoritmo voraz
Sean S una solución parcial, E el conjunto factible de elementos que pueden ser añadidos a la solución parcial, e_i los elementos del conjunto E y g una función voraz. $S \leftarrow \emptyset$ $mientras\ (S\ no\ factible)\ hacer$ $Evaluar\ la\ función\ voraz\ g\ para\ cada\ elemento\ de\ E$ $Seleccionar\ el\ mejor\ elemento\ e$ $\in E\ de\ acuerdo\ con\ el\ valor\ de\ la\ función\ voraz\ g$ $(i.e.\ e = \arg\min\{g(e_i)\ \forall\ i = 1, \dots, \lvert E \rvert\})$ $Actualizar\ E$ $fin\ mientras$

Figura 3: Pseudocódigo del algoritmo voraz.

Fase de Mejora

La segunda fase de GRASP corresponde al proceso de búsqueda local. Los algoritmos propuestos en esta fase varían desde algoritmos sencillos de búsqueda local a técnicas más avanzadas y eficientes.

Algoritmo de búsqueda local
Sea $f(x)$ la función a minimizar, x una solución factible y $N(x)$ una estructura de vecindad. $CriterioParada \leftarrow falso$ $mientras\ (no\ se\ satisfaga\ CriterioParada)\ hacer$ $x' = \arg\min\{f(y): y \in N(x)\}$ $si\ \big(f(x') < f(x)\big) entonces$ $x \leftarrow x'$ $en\ otro\ caso$ $CriterioParada \leftarrow cierto$ $fin\ si$ $fin\ mientras$

Figura 4: Pseudocódigo del algoritmo de búsqueda local.

CAPÍTULO 5

5. OPTIMIZACIÓN DE RUTAS DE RECOGIDA DE RESIDUOS EN ZONAS MIXTAS URBANAS-RURALES Y OROGRAFÍA SINGULAR

En este capítulo se realiza un análisis del problema, que incluye qué sucede en los problemas reales de recogida de residuos y los detalles específicos del problema. A continuación se describen los procedimientos que se van a utilizar para llevar a cabo este proyecto, incluyendo la función de costes que se va a aplicar.

5.1 ANÁLISIS DEL PROBLEMA

La raíz de este proyecto surge de una problemática que sucede en la vida real. Actualmente hay municipios que se encuentran sin una recogida de residuos completamente implementada, no hay puntos concretos de recogida de residuos a los que los camiones tengan que ir a recoger, sino que las personas dejan la basura en las propias calles. Por esa razón, en vez de tener que realizar rutas para los camiones de forma que tengan que recorrer todos los puntos, tienen que pasar por todas y cada una de las calles donde hay viviendas para recoger la basura. Esto complica el problema, además de alargar la complejidad del propio trabajo. Hay que recorrer muchísima más distancia, hay calles a las que los camiones no pueden acceder en la vida real porque no cabe el propio camión,…

En territorios de origen volcánico, como las Islas Canarias, la orografía es abrupta y compleja, con muchos cambios de altura. En el caso de Tenerife, toda la isla "desciende" del propio volcán, por lo que es todo en pendiente hasta llegar a la costa. Esto provoca que en la mayor parte de municipios la orografía sea en pendiente. Esta orografía provoca grandes dificultades para el transporte. Por ejemplo, se ha comprobado que se produce un mayor desgaste de los vehículos, menor duración de los mismos. La vida media de un vehículo en el resto de España asciende a 16.7 años, siendo en Canarias notablemente inferior. Además, este problema se acentúa en los vehículos de recogida de residuos necesarios en este estudio. Este tipo de vehículos necesitan no sólo tener la capacidad de recogida adecuada, sino ser capaces de subir por todas esas calles inclinadas para realizar su labor. Por otra parte, hay gran cantidad de barrancos y desniveles provocados precisamente por el origen volcánico de las islas. Estos barrancos provocan que pases a haber poca distancia entre dos

puntos, haya que realizar un rodeo bastante grande a causa de un barranco o similar. Este problema provoca un incremento en la distancia entre diferentes puntos aparentemente cercanos, además de suponer un enorme riesgo por el rápido deterioro de las carreteras debido a la inestabilidad del terreno.

La investigación se centra en los problemas de recogida de residuos en municipios con zonas rurales y dispersas con las características anteriores. Los vehículos que realizan la recogida tienen que recorrer mucha distancia por calles poco adaptadas. Por esa razón, los camiones de recogida de residuos han de recorrer el municipio comenzando la ruta por las partes más altas, de forma que según se vayan cargando de residuos, vayan descendiendo por el municipio. Con eso se consigue reducir enormemente el esfuerzo que realiza el coche, con el consiguiente ahorro en combustible y en elementos de desgaste del vehículo.

Las restricciones vienen impuestas por los recursos disponibles, el principal es la flota de vehículos disponible, que suele estar formada por un número de camiones determinado o fijo. La principal característica que hay que tener en cuenta con los vehículos es la carga máxima que pueden almacenar. Esto determinará no sólo el coste de la ruta sino la capacidad de carga máxima.

Las vías por las que hay que circular son bastante problemáticas, sobre todo por la inclinación de las mismas. Esta inclinación hace que suponga un coste enorme recorrerlas de manera equivocada, de forma ascendente en vez de descendente. Es un dato decisivo en este problema, ya que sale más rentable que los vehículos partan de la zona más elevada del municipio y vayan recorriendo las calles de manera descendente.

Existen algunos problemas de similares características y restricciones, pero no son contemplados en este proyecto. En algunos problemas se tienen en cuenta los horarios de los trabajadores, de forma que se tiene en cuenta el tiempo que se tarda en recorrer cada ruta para que no exceda la jornada laboral del trabajador. Además, este tipo de referencias sirven para equilibrar las rutas obtenidas, ya que podría darse el caso de que un trabajador cumpliera su jornada laboral pero a los demás les sobrase tiempo, o que un trabajador cargase con la mayoría de los residuos y los demás trabajadores casi no cargasen residuos,…

5.2 DESCRIPCIÓN DE LA FUNCIÓN DE COSTE

En general, a la hora de resolver este problema hay que tener en mente una serie de objetivos, que se resumen en minimizar. Hay que tener en cuenta los objetivos de las organizaciones y clientes, asociados con los objetivos de eficiencia y eficacia del servicio. El objetivo de estos problemas es optimizar los costes expresados en reducción o minimización de distancia, tiempo, coste,… Normalmente el coste de

una ruta se basa fundamentalmente en la distancia. Pero para este problema concreto hay que recoger características específicas para minimizar el coste, a parte de la carga que tiene el vehículo en cada momento, hay que tener en cuenta la inclinación de la calle por la que está circulando. Para ello se ha desarrollado una función que devuelve el coste calculándolo a partir de la distancia de la calle por la que se va a circular, su porcentaje de inclinación multiplicado por dicha distancia, y añadiéndole el producto del porcentaje de carga en ese momento y la distancia. De esta forma, si el vehículo opta por ir en calles descendentes, el factor de inclinación jugará en su favor facilitándole el descenso y reduciendo el coste. Por el contrario, en los casos en los que tenga que circular por una calle en sentido ascendente se incrementará el coste en función de la inclinación de la misma y lo cargado que vaya en ese momento. Este factor de inclinación multiplica la distancia total a recorrer en esa calle.

Además, otro dato a tener en cuenta es la carga en cada momento del vehículo. No es lo mismo que un vehículo vaya sin carga a que vaya prácticamente lleno, ya que cuanto más carga lleve el camión, más coste supone el recorrido. Por esa razón, también se ha obtenido un factor de carga resultado de multiplicar el porcentaje de llenado en cada momento del camión a la distancia que ha de recorrer en esa calle.

Por consiguiente, para calcular el coste de una ruta, se ha utilizado esta fórmula:

$$d_{ij} + \left(d_{ij} * incl_{ij}\right) + \left(d_{ij} * (C_{acumulada}/C_{total})\right)$$

Siendo d_{ij} la distancia existente entre el arco i – j, $incl_{ij}$ se corresponde con la inclinación que tiene el arco i – j. La carga del vehículo viene representada en esta fórmula por $C_{acumulada}$ como la carga acumulada del vehículo hasta ese punto, y C_{total} como la carga total de que dispone ese vehículo. De esta forma no sólo hay una referencia en distancia del recorrido como aparece usualmente en este tipo de problemas estudiados, sino que se puede hacer que influya de forma decisiva en el coste tanto la inclinación de la calle a recorrer como la carga acumulada hasta ese momento por el vehículo.

Así pues, el resultado de este nuevo coste se obtendría de sumarle a la distancia de la calle el factor de inclinación y el factor de carga obtenido. Gracias a que el recorrido se hace de arriba hacia abajo, el coste puede llegar a ser inferior que si tan solo se aplicase un algoritmo GRASP sin tener en cuenta nada más que la distancia para calcular el coste total de la ruta.

A modo de mejora de la fórmula se podría afinar más incluyendo para cada vehículo del modelo concreto que se esté estudiando el consumo medio. Así se podría realizar un cálculo del consumo a través del consumo medio en llano o en pendientes de dicho vehículo.

CAPÍTULO 6

6. DESARROLLO E IMPLEMENTACIÓN DEL GRASP

En este capítulo se describe cómo ha sido el desarrollo y la implementación del procedimiento propuesto. Esta descripción abarca desde el comienzo de la implementación hasta la descripción de cada una de las clases del programa. Además, en un segundo apartado se describen los métodos de optimización utilizados.

6.1 ESPECIFICACIÓN DE CLASES

A la hora de comenzar a implementar el código se ha realizado un estudio intensivo para evaluar cuál sería la estructura óptima del código. Para ello se ha tenido en cuenta el planteamiento del problema.

Un municipio rural con orografía compleja requiere una organización en el servicio de recogida de residuos. Este municipio está completamente inclinado, pudiendo distinguir una parte alta y una parte baja dentro del propio municipio. En la parte más baja de dicho municipio se encuentra el depósito de basuras, punto al que hay que llegar con la basura de todo el municipio. Teniendo en cuenta esta inclinación, es conveniente comenzar las rutas de los vehículos de recogida de residuos en la parte más alta del municipio, de forma que al ir bajando hasta el depósito sea más fácil. Si las rutas comenzasen en el depósito, dado que está en la parte más baja, tendrían que subir las calles cargados con la basura que van recogiendo, con el coste añadido que eso conlleva.

Así pues, ya se pueden distinguir las diferentes clases que van a tomar parte en la implementación de este problema. Se propone una clase vehículo, que almacenase todas las características de la solución de este problema tales como la matrícula, la capacidad,… Dado que la empresa de recogida de residuos tiene más de un vehículo, es necesario que se implemente una clase flota, que almacene un vector con cada uno de los vehículos disponibles, además de los procedimientos oportunos para acceder a dichos datos.

Por tanto, ante la dificultad de distinguir las calles se plantearon dos alternativas. Está la posibilidad de distinguir las calles en sí, guardarlas en una clase donde se

almacenase su tamaño, su inclinación y la demanda de recogida que tenía. El problema de eso es que se complicaba el hecho de realizar la ruta, no se relacionaban muy bien entre sí las calles como para realizar un recorrido. Entonces se crea una matriz de adyacencia para dichos arcos, de forma que se sepa qué arcos son adyacentes entre sí. El problema que genera eso, es que no sólo hay que tener en cuenta qué calles están conectadas entre sí, sino por qué extremo. Esto es, si se empieza en una calle tengo y se recorre, se tiene que saber por qué calle hay que ir a continuación. No sirve de nada que al principio de la calle pudiera haber otra que enlace, hay que saber por qué extremo están unidas.

Posteriormente se identifican cada una de las intersecciones de las calles, es decir, los vértices. Cada vértice tiene un id, y ese id está conectado con otro id a través de una matriz de adyacencia. Además, se incluye una matriz de distancias que almacene las distancias que hay entre cada punto y su adyacente, marcando como distancia infinita o no alcanzable todos aquellos arcos no existentes en el problema.

La inclinación y la demanda de cada arco se almacenan de la misma manera que las distancias, ya que son características a tener en cuenta por esa calle. Por tanto, se han identificado cada uno de los vértices como puntos a visitar o clientes, y se ha creado una clase clientela que almacena todos esos clientes. De cada cliente se dispone de su vector de adyacencia, con los puntos que son adyacentes a ese cliente. Además, se almacenan las anteriormente mencionadas matrices de distancias, inclinación y demandas.

Para poder obtener los datos, se ha determinado que se lean de un fichero de entrada, de forma que sea más fácil introducir los datos de prueba para el problema. Así que se implementa una clase que realiza la lectura de fichero. Este fichero tiene un formato estándar que consta de: nombre del fichero, número de vehículos, capacidad de cada vehículo, id de cada cliente, coordenada x e y, adyacentes de cada cliente, la demanda de cada par de adyacentes y su inclinación. Como alternativa, también es posible pasarle al programa las distancias de cada punto a su adyacente en vez de sus coordenadas. Esto es muy útil cuando en vez de coordenadas ejemplo de un problema de las cuales se calcula una distancia euclídea, lo que queremos sea un ejemplo de la vida real. Así, podremos introducir los puntos de intersección de las calles y la distancia real que hay entre dos puntos, que para nada se correspondería con la distancia euclídea.

Para la generación de ruta se implementa la clase utilizando el algoritmo GRASP. Este realiza la ruta eligiendo para cada punto, de entre sus tres puntos más cercanos, uno aleatoriamente. Esto es así para que no se quede en óptimos locales, sino que busque más allá. El algoritmo implementado comienza en los puntos iniciales, que se corresponden por estándar con los puntos con el mismo id que el vehículo (el vehículo 1 comienza en el punto 1, el vehículo 2 comienza en el punto 2,…). A continuación se cogen los tres adyacentes más cercanos a ese punto y se elige aleatoriamente uno de los tres. Los puntos que se van visitando se guardan en una

matriz, donde cada una de las filas termina siendo la ruta de cada vehículo. El algoritmo deja de generar la ruta cuando ha llegado al depósito y pasa a generar la ruta del siguiente vehículo, evitando pasar por los puntos ya visitados por anteriores vehículos.

Cuando finaliza la ejecución del algoritmo, el programa habrá obtenido tres rutas que comienzan en la parte más alta del municipio y terminan en el depósito. Al finalizar tendremos las rutas de cada uno de los vehículos con todos los puntos ya visitados.

Descripción de las clases

- Clase cliente: Esta clase representa a un cliente concreto. En ella se almacenan los atributos privados id o identificador de cliente, coordenada x del punto, coordenada y del punto, el número de adyacentes que tiene dicho cliente, el vector de adyacentes, el vector de demandas y el vector de inclinaciones. Los métodos de esta clase son públicos y se basan en un constructor que inicializa todos los atributos, y getter para consultar datos concretos del cliente cuando sea necesario.
- Clase clientela: Dado que tenemos un número a priori desconocido de clientes, esta clase los almacena todos en un vector dinámico. También se almacena en esta clase la matriz de distancias que almacena la distancia desde un cliente al resto (en caso de que un cliente no sea adyacente a otro su valor en la matriz de distancias será infinito). La función de esta clase es devolver información acerca de los clientes, de las demandas de dichos clientes, la distancia entre ellos, la inclinación de cada arco entre dos puntos adyacentes,…
- Clase vehículo: Para cada vehículo se crea un elemento de clase vehículo. De él se almacena el id o identificador y su capacidad. En caso de que se quisieran añadir ventanas de tiempo sería cuestión de añadirle un atributo que indicase la jornada laboral. En el cliente también sería necesario incluir cuánto tiempo es necesario para recoger los residuos de cada arista y se representaría a través de una matriz de tiempos similar a la matriz de distancias, en la que sólo los puntos adyacentes tienen un valor. Los métodos de esta clase son getter y setter.
- Clase flota: Almacena el total de vehículos disponibles en un vector dinámico. Tiene una función similar a la de la clase clientela pero aplicada en este caso a los vehículos en vez de a los clientes. Sus métodos son básicamente getter y setter.
- Clase lectura: A través de esta clase, con el nombre del fichero de lectura como atributo privado, se procede a la lectura del fichero de entrada en el formato establecido. A lo largo de la lectura del fichero se van creando e insertando clientes y vehículos que se incluirán en las clases clientela y flota respectivamente.

- Clase ruta: Esta clase es la que ejecuta el algoritmo metaheurístico GRASP. Entre todos los métodos de esta clase se van generando las rutas paso a paso. Dispone de una función que genera las rutas, que se sirve de funciones auxiliares fundamentales para la ejecución correcta del código como pueden ser la clase que devuelve los adyacentes, otra clase que ordena dichos adyacentes y elige uno aleatorio entre los 3 con menor distancia o menor coste (en función de si se quiere tener en cuenta la fórmula del coste o sólo la distancia),… además de los getter y setter oportunos.
- Clase simplificar: Cuando se han ejecutado los métodos de las otras clases, es posible simplificar la ruta resultante de varias maneras. Una de ellas puede ser iterando un mayor número de veces e ir quedándonos con la mejor de las soluciones. Otra opción es simplificar las rutas con el fin de quitar caminos sobrantes o redundantes.
- Clase escritura: Por último, esta clase se encarga de almacenar en un fichero de salida, con el formato previamente establecido, las rutas obtenidas junto con sus costes, distancias, carga,…

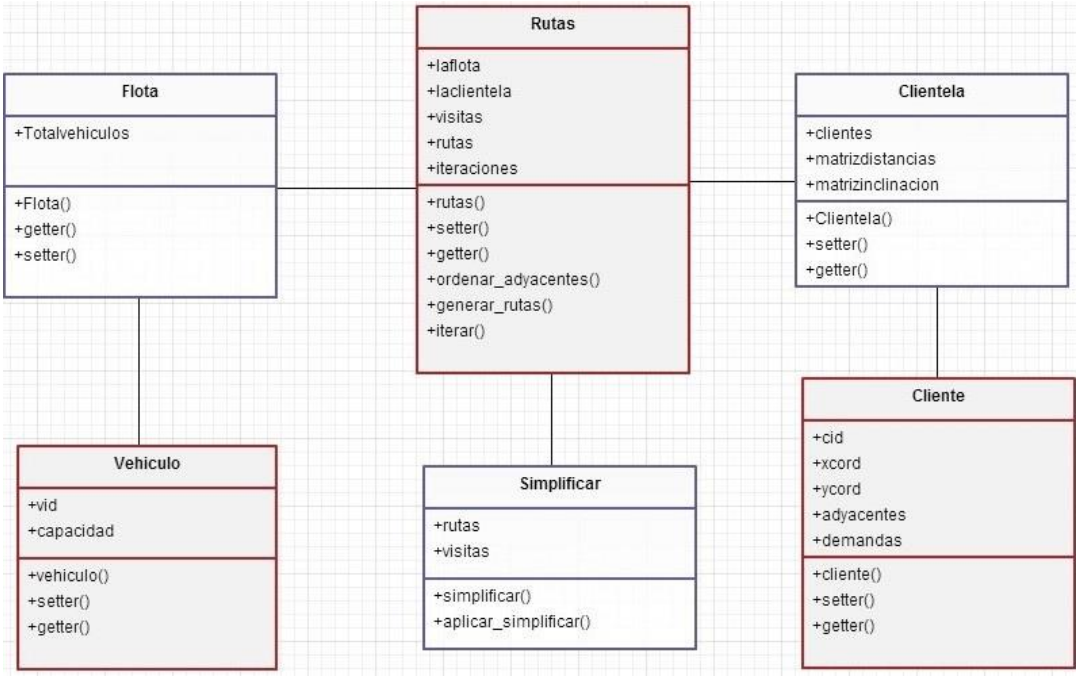

Figura 5: Diagrama UML de nuestro programa.

6.2 ESPECIFICACIÓN DEL PROCEDIMIENTO

La primera parte del programa se encarga de introducir los datos desde el fichero de entrada indicado por el usuario. Este fichero contiene los datos de los vehículos implicados en el problema, además de los datos de todos los puntos. La información acerca de los vehículos incluye el número de vehículos que hay que utilizar en el problema y su capacidad máxima. En este caso, con el fin de asemejarse lo más posible al problema de Icod de los Vinos, se van a realizar una ruta para cada vehículo disponible. Cada uno de los puntos incluye la información de sus coordenadas euclídeas, de sus adyacentes, de la inclinación de los arcos adyacentes y de su demanda. Como alternativa, el programa también admite que se introduzca una matriz de distancias, ya que para los casos reales la distancia euclídea no sirve. A continuación el programa genera la matriz de distancias, incluyendo valores infinitos para los arcos no existentes en función de la matriz de adyacencia, la matriz de demandas y la de inclinación.

El siguiente paso del programa es el de generar las rutas. Las rutas son generadas a través de una implementación del algoritmo metaheurístico GRASP. Este algoritmo se dividía en dos partes fundamentales: un procedimiento voraz que construye una solución inicial, y una mejora de la solución construida. Se empieza en el punto de partida del primer vehículo y se busca de entre los posibles adyacentes cuáles son los más cercanos ya sea en distancia o en coste. Una vez son identificados esos puntos, se elige aleatoriamente uno de los tres con menor distancia. A continuación se marca como visitado con el fin de que no pase por ahí otro vehículo. Para marcarlo como visitado se dispone de un vector de booleanos, que comienza inicializado a false ya que no ha sido visitado ningún punto. Según se va avanzando en el proceso se van marcando puntos a true ya que son visitados, y con eso se consigue que en posteriores rutas no se vuelvan a repetir. Después se vuelve a realizar todo este procedimiento para el siguiente punto que hemos añadido a la ruta.

Mientras no se hayan visitado todos los puntos:

- Paso 1: se ordenan los puntos candidatos a ser visitados en función del coste.
- Paso 2: Se selecciona aleatoriamente uno de los tres elementos con menor coste.
- Paso 3: Se marca como visitado y se inserta en la ruta.

Si a lo largo del recorrido un vehículo se queda atascado por no tener puntos adyacentes sin visitar, entonces el vehículo irá hacia atrás en la ruta hasta encontrarse con un punto que tenga adyacentes sin visitar. Una vez genera la ruta de un vehículo al llegar al depósito, procede a generar las de los demás vehículos evitando pasar por los lugares ya visitados. Cuando están finalizadas las rutas para cada vehículo, esas rutas no van a contener todos los puntos, ya que quedarán sin visitar aquellos que tengan alguna restricción particular como por ejemplo una calle cortada o una calle

por la que no puede pasar un vehículo físicamente. Con el fin de optimizar estas rutas obtenidas, e incluso minimizar los nodos que quedan sin visitar, este primer paso se realiza un número determinado de iteraciones indicado previamente por el usuario. Dado que de cara a la carga del vehículo es necesario tener en cuenta todos esos putos que no han sido visitados, a través de un nuevo procedimiento inserta los nodos en las rutas. Para insertar cada uno de los nodos, el procedimiento identifica cuáles son los adyacentes del nodo, los ordena en función del coste y elige aleatoriamente uno de los tres más cercanos. Posteriormente se busca en las rutas dónde se encuentra ese nodo más cercano y, en caso de que no incumpla las restricciones de carga del problema, será insertado en el lugar correspondiente. En caso de que incumpla las restricciones de carga, se insertará en otro de sus nodos adyacentes.

Por último se busca optimizar las rutas con diferentes procedimientos tales como la simplificación. Cuando se generan las rutas y cuando se insertan esos nodos sin visitar pueden producirse giros o redundancias. Con el fin de resolver este inconveniente, se ha implementado un procedimiento que las simplifica. Se dispone de un vector de booleanos que almacena si un punto ha sido visitado o no (inicializado a false). El procedimiento comienza en el elemento de la ruta *a*, y comprueba si el elemento *a+1* ha sido visitado. En caso afirmativo se pueden dar tres casos:

- Que el elemento a sea igual al elemento *a+2*, entonces se eliminan de la ruta los elementos *a+1* y *a+2*.
- Que el elemento a sea adyacente al elemento *a+2*, entonces se elimina el elemento *a+1* dado que ya ha sido visitado.
- En caso contrario no se haría nada.

Este proceso se realiza con todos los puntos de la ruta. En caso de que se produzca alguna eliminación, habrá un booleano que se ponga a true para indicar que se ha producido. Esto es necesario ya que al eliminar algunos elementos se siguen produciendo redundancias, así que se sigue ejecutando el procedimiento mientras se produzcan eliminaciones. Cada vez que se vuelve a ejecutar esta simplificación, todas las casillas del vector de visitados vuelven a false.

6.3 ESPECIFICACIÓN DE LA ENTRADA-SALIDA

Para una correcta ejecución del código se ha implementado un sistema a través del cual, desde consola de comandos se introducen todos los datos necesarios para la ejecución del mismo. De esta forma, se introduce el nombre del ejecutable y el fichero de entrada. Opcionalmente se puede indicar el nombre del fichero de salida donde queremos que se guarden las rutas resultantes e incluso el número de

iteraciones que queremos que ejecute el programa con el fin de obtener una ruta lo más optimizada posible.

$$< nombre\_ejecutable >< entrada >< salida >< iteraciones >$$

Al finalizar la ejecución, la ruta generada para cada uno de los vehículos se almacenará en un fichero de salida (ya sea el indicado por el usuario o un nombre por defecto). El formato de representación en el fichero de salida viene dado por la ruta para ese vehículo, que finaliza siempre en el depósito, a continuación la carga con la que termina el vehículo la ruta, el coste que ha supuesto la ruta teniendo en cuenta el factor de inclinación para cada calle y la carga que llevaba en cada momento, y la distancia total recorrida.

$$Ruta; Carga\_de\_vehiculo; Coste\_ruta; Distancia\_recorrida$$

CAPÍTULO 7

7. EXPERIMENTACIÓN

En este capítulo se describe la experimentación elaborada. Para la realización de estas pruebas hemos utilizado una instancia que ejemplifica las características del problema real de Icod de los Vinos.

Para la implementación de este programa se ha realizado en el lenguaje C++. Se ha utilizado la herramienta Eclipse aprovechando además su opción de vincularla al sistema de control de versiones Git. El equipo donde ha sido implementado este proyecto y en el cual se han hecho todas las pruebas y la experimentación tiene las siguientes características técnicas:

Procesador	Intel® Core™ i7 – 4702MQ (2.2 GHz / 3.2 GHz w/ Turbo Boost)
Memoria RAM	8 GB DDR3
Almacenamiento	256 GB SSD, 750 GB SATA
Tarjeta gráfica	NVIDIA GeForce GTX760M w/ 2 GB GDDR5
Pantalla	17.3''
Sistema operativo	Windows 8.1 Pro x64

Tabla 4: Descripción del equipo en el que se realizaron las pruebas.

A continuación se muestra un ejemplo a través del cual se busca ilustrar mejor el funcionamiento del programa y los resultados que se obtienen del mismo. Para la realización de las pruebas hemos utilizado una instancia que ejemplifica las características del problema real de Icod de los Vinos, a través del cual surgió este proyecto escogiendo una instancia semejante con los mismos objetivos y restricciones. Tal y como se expone en el apartado anterior, para la resolución de este problema será fundamental la señalización de las los vértices de la red viaria, tal y como se muestra en la figura de muestra. El fichero de entrada de este ejemplo contiene el número de vehículos que se van a utilizar para generar las rutas, y el conjunto de vértices que hay que recorrer junto con sus coordeandas x e y, sus adyacentes, y la demanda e inclinación de los arcos entre ese punto y cada uno de sus adyacentes. Con el fin de que este ejemplo se asemeje en la medida de lo posible

al caso de Icod de los Vinos, se ha determinado que este ejemplo disponga de tres vehículos y que se quiera buscar una ruta para cada uno de ellos. Estos tres vehículos tienen una carga homogénea, siendo su carga máxima 220 unidades. Además, los puntos están ordenados de forma que los números más pequeños son los que se encuentran a más altura en el plano. De esta forma, cuando se comience a ejecutar el algoritmo de generación de rutas, cada vehículo empezará en el nodo con su mismo identificador. Esto es, el vehículo con identificador 1 comenzará a recorrer la ruta desde el vértice con identificador 1, y así sucesivamente. Cada una de las rutas ha de terminar en el depósito o punto 0. En el fichero de entrada viene especificado que cada uno de los puntos que se encuentren en la misma calle que el depósito o punto 0, son todos adyacentes de 0. Por esa razón, en algunas ocasiones las rutas puede parecer que se saltan puntos de esa calle, pero se debe a que el 0 es adyacente de todos ellos.

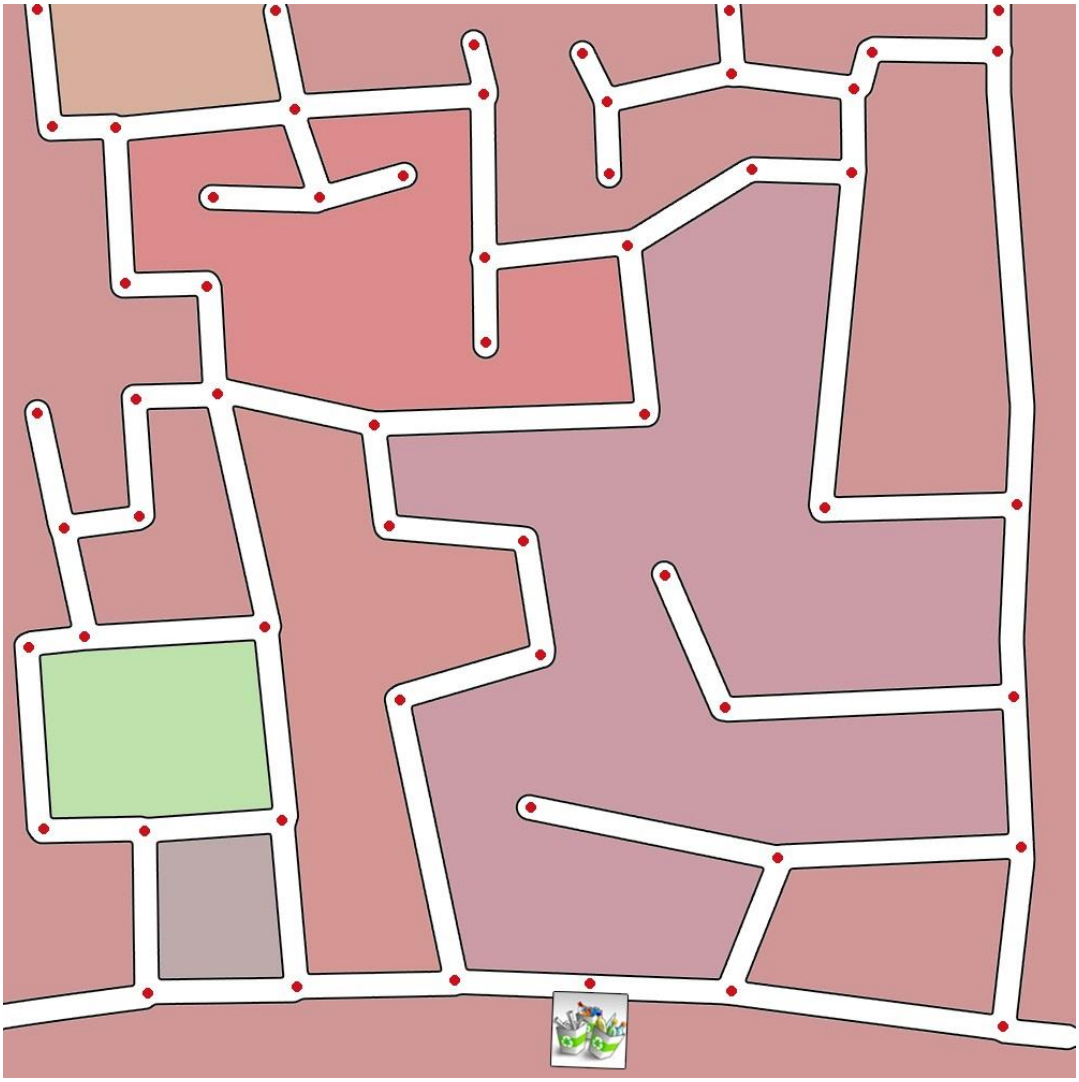

Figura 6: Mapa del municipio con los vértices señalados.

Dadas las dimensiones de este problema, se hace muy engorroso mostrar la matriz de distancias con los 57 puntos. Por ello, se hará uso del Anexo I para mostrar tanto la matriz de distancias como la matriz de inclinación de cada nodo con su adyacente. Con el fin de representar los nodos entre los que no existía ningún arco, se hará uso de una constante "infinito" que dará un valor concreto muy grande que el programa descartará cuando lo encuentre. La matriz de distancias de cada fichero de entrada viene determinada por la adyacencia entre los diferentes nodos. Si entre dos nodos adyacentes, *a* y *b*, en la vida real hay una calle de doble sentido, el valor de la distancia será el mismo de *a* hacia *b* que de *b* hacia *a*. Por el contrario, si la calle es de un único sentido, la distancia tendrá un valor en un sentido, pero en el sentido contrario será "infinito".

	0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*
0	0	999	999	999	999	999	999	999	999	999	999	999	999	999	999
1	999	0	999	999	999	4.0	999	999	999	999	999	999	999	999	999
2	999	999	0	999	999	999	999	4.0	999	999	999	999	999	999	999
3	999	999	999	0	999	999	999	999	999	999	999	999	4.0	999	999
4	999	999	999	999	0	999	999	999	999	999	999	999	999	999	3.0
5	999	4.0	999	999	999	0	1.0	999	999	999	999	999	999	999	999
6	999	999	999	999	999	1.0	0	5.0	999	999	999	999	999	999	999
7	999	999	4.0	999	999	999	5.0	0	999	5.0	999	999	999	999	999
8	999	999	999	999	999	999	999	999	0	2.0	999	999	999	999	999
9	999	999	999	999	999	999	999	5.0	2.0	0	999	999	999	999	999
10	999	999	999	999	999	999	999	999	999	999	0	1.0	999	999	999
11	999	999	999	999	999	999	999	999	999	999	1.0	0	2.0	999	999
12	999	999	999	4.0	999	999	999	999	999	999	999	2.0	0	999	999
13	999	999	999	999	999	999	999	999	999	999	999	999	999	0	4.0
14	999	999	999	999	3.0	999	999	999	999	999	999	999	999	4.0	0

Tabla 2: Fragmento ejemplo de la matriz de distancias.

En el caso de la matriz de inclinación, la forma de establecer si recorremos una calle en sentido ascendente o descendente viene dada por el factor de inclinación de la misma. En caso de que el valor sea negativo, quiere decir que estamos recorriendo la calle en sentido descendente, por lo que no sólo no supone mayor esfuerzo recorrerla sino que es mucho más fácil incluso que si no tuviera inclinación alguna. Si al contrario este valor es positivo, quiere decir que estamos recorriendo la calle en sentido ascendente, lo que supondrá un notable incremento en el coste, sobre todo cuánto más cargado esté el camión. Si una arista tiene doble sentido, el valor en el sentido ascendente de la arista será positivo y en sentido descendente será negativo.

Cuando en una determinada arista no hay ningún tipo de inclinación, el valor en la matriz será 0 en ambos sentidos. En el caso de que una arista tenga un solo sentido, el valor de la inclinación sólo constará con el signo y en el sentido correspondiente, y en el sentido contrario tendrá un valor "infinito".

	0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*
0	0	999	999	999	999	999	999	999	999	999	999	999	999	999	999
1	999	0	999	999	999	-0.1	999	999	999	999	999	999	999	999	999
2	999	999	0	999	999	999	999	-0.2	999	999	999	999	999	999	999
3	999	999	999	0	999	999	999	999	999	999	999	999	-0.3	999	999
4	999	999	999	999	0	999	999	999	999	999	999	999	999	999	-0.1
5	999	+0.1	999	999	999	0	0	999	999	999	999	999	999	999	999
6	999	999	999	999	999	0	0	0	999	999	999	999	999	999	999
7	999	999	+0.2	999	999	999	0	0	999	0	999	999	999	999	999
8	999	999	999	999	999	999	999	999	0	-0.3	999	999	999	999	999
9	999	999	999	999	999	999	999	0	+0.3	0	999	999	999	999	999
10	999	999	999	999	999	999	999	999	999	999	0	-0.1	999	999	999
11	999	999	999	999	999	999	999	999	999	999	+0.1	0	0	999	999
12	999	999	999	+0.3	999	999	999	999	999	999	999	0	0	999	999
13	999	999	999	999	999	999	999	999	999	999	999	999	999	0	0
14	999	999	999	999	+0.1	999	999	999	999	999	999	999	999	0	0

Tabla 3: Fragmento ejemplo de la matriz de inclinación.

7.1 RESULTADOS

Tal y como se menciona en el capítulo anterior, a la hora de generar en la primera parte del problema pude comprobar que en las rutas se quedaban puntos sin visitar correspondientes a calles cortadas o inaccesibles por los vehículos, pero que hay que tenerlas en cuenta ya que hay que cubrir su demanda. Estos puntos se incluyen en la ruta con el fin de tenerlos en cuenta de cara a la carga del vehículo. Para ilustrar la eficacia de realizar iteraciones, será utilizada la instancia descrita anteriormente. En esta tabla se muestran los resultados obtenidos en función de la cantidad de iteraciones realizadas. Los resultados vienen dados por el coste que supondría a la flota de vehículos realizar las rutas, por la distancia real que recorren y el tiempo total de ejecución del programa. El coste se calcula teniendo en cuenta la longitud del arco, el factor de inclinación y el factor de carga del vehículo. La distancia real que recorren ilustra también cuál sería el coste en caso de que no se tuviera en cuenta ningún tipo de factor adicional para su cálculo. El tiempo de ejecución está medido en segundos.

Nº iteraciones	Coste	Distancia	Tiempo de ejecución
1	297.1	310	0.006346
5	291.8	304	0,022929
10	281.4	294	0,026995
50	299.5	312	0,067834
100	291.8	304	0,137
500	284.1	296	0,623642
1000	281.4	294	1,20705
5000	281.4	294	6,24899
10000	276.6	290	12,6299
50000	275.9	290	62,7447
100000	266.2	280	127,086

Tabla 4: Resultados en función del número de iteraciones.

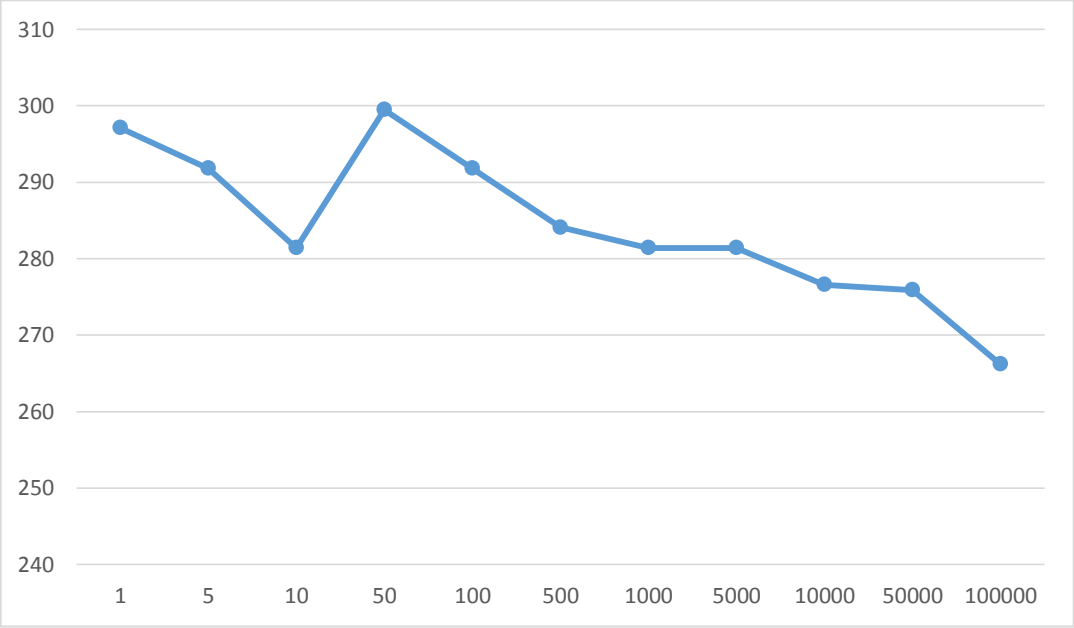

Figura 7: Gráfica del coste en función del número de iteraciones.

Gracias a la figura 7 podemos observar cómo los costes van disminuyendo, es decir, mejorando, en función del número de iteraciones. Cuantas más iteraciones ejecuta el programa, mejores resultados obtiene ya que evalúa mayor cantidad de posibilidades.

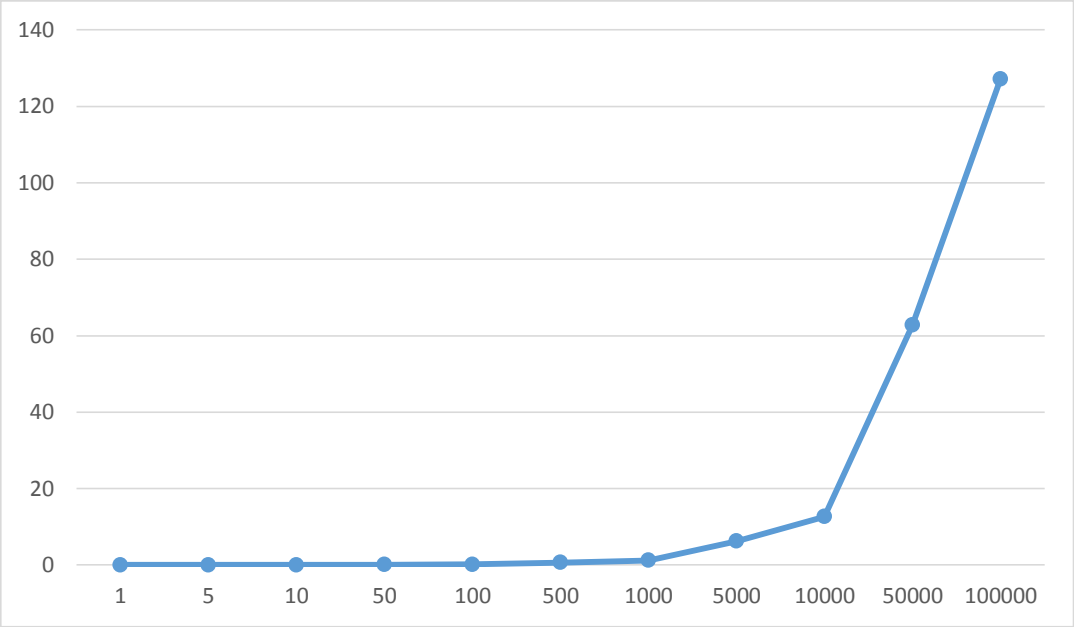

Figura 8: Gráfica del tiempo de ejecución en función del número de iteraciones.

A través de la figura 8 podemos comprobar como el tiempo de ejecución se mantiene bastante estable hasta las diez mil iteraciones, desde donde empieza a incrementarse el tiempo notablemente. A pesar de ese incremento en el tiempo, sigue sin ser significativo y no supone una dificultad ya que se obtienen mejores resultados.

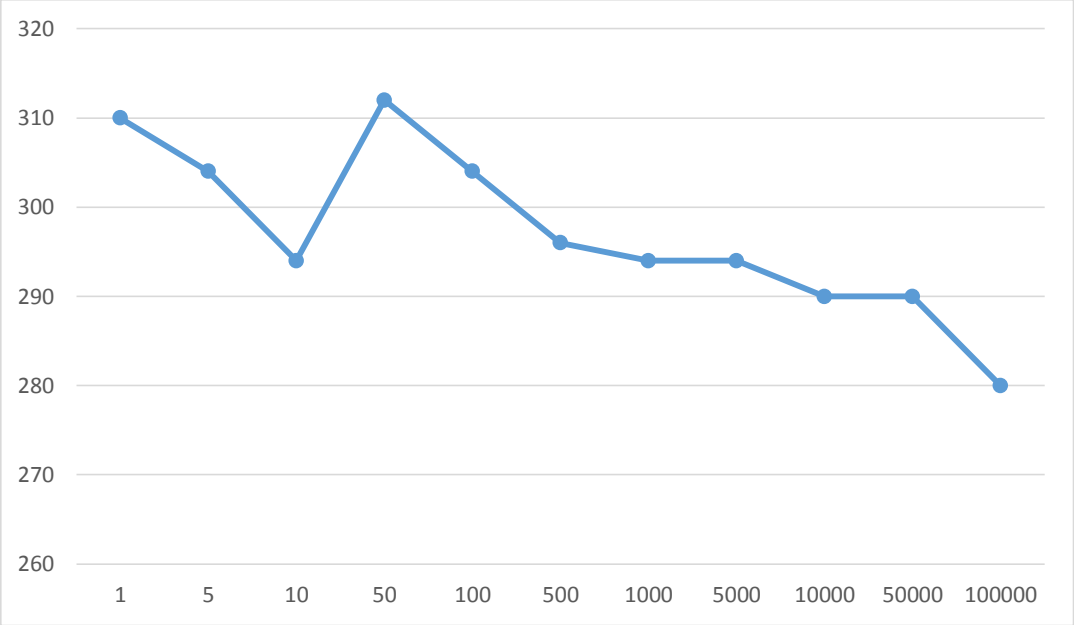

Figura 9: Gráfica de la distancia en función del número de iteraciones.

A continuación se listan unas tablas que muestran el comportamiento del programa en función del número de iteraciones. En una primera parte se listan las rutas obtenidas tras ese número de iteraciones indicado, incluyendo los nodos que han quedado sin visitar. A continuación se muestran las rutas parciales con los nodos sin visitar ya insertados en las rutas. Por último se incluyen las rutas finales, es decir, tras haber pasado por el procedimiento de simplificación. Junto con estas rutas finales se incluye el coste de cada una de ellas, el coste total y el tiempo que ha llevado la ejecución total del programa.

Nº de iteraciones	1	
Ruta vehículo 1	1 5 6 15 23 30 38 47 52 0	
Ruta vehículo 2	2 7 17 18 17 16 17 7 9 24 25 24 26 32 31 40 41 42 39 53 0	
Ruta vehículo 3	3 12 11 19 11 10 11 12 56 21 20 21 34 35 44 43 33 43 44 50 55 0	
Puntos sin visitar	4 8 13 14 22 27 28 29 36 37 45 46 48 49 51 54	
Ruta vehículo 1	1 5 6 15 23 30 28 29 28 30 38 37 36 45 36 37 27 22 27 37 36 45 36 37 38 47 46 51 46 47 52 0	
Ruta vehículo 2	2 7 17 18 17 16 17 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
Ruta vehículo 3	3 12 11 19 11 10 11 12 56 13 56 21 20 21 34 35 14 4 14 35 44 43 33 43 44 50 49 48 49 54 49 48 49 50 55 0	
Ruta final vehículo 1	1 5 6 15 23 30 28 29 28 30 38 37 36 45 36 37 27 22 27 37 38 47 46 51 46 47 52 0	
	Coste de la ruta	91.6
	Distancia recorrida	95
Ruta final vehículo 2	2 7 17 18 17 16 17 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
	Coste de la ruta	79.8
	Distancia recorrida	85
Ruta final vehículo 3	3 12 11 19 11 10 11 12 56 13 56 21 20 21 34 35 14 4 14 35 44 43 33 43 44 50 49 48 49 54 49 50 55 0	
	Coste de la ruta	125.7
	Distancia recorrida	130
Coste total de las rutas	297.1	
Distancia total	310	
Tiempo (s)	0.006346	

Tabla 5: Pruebas del fichero de entrada ejemplo con una iteración.

Nº de iteraciones	10	
Ruta vehículo 1	1 5 6 15 23 30 28 29 27 22 27 37 36 45 46 47 52 0	
Ruta vehículo 2	2 7 17 18 17 16 17 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
Ruta vehículo 3	3 12 11 19 11 10 11 12 56 21 20 21 34 35 44 43 33 43 44 50 49 48 49 54 0	
Puntos sin visitar	4 13 14 38 51 55	
Ruta vehículo 1	1 5 6 15 23 30 28 29 27 22 27 37 36 45 46 51 46 47 38 47 52 0	
Ruta vehículo 2	2 7 17 18 17 16 17 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
Ruta vehículo 3	3 12 11 19 11 10 11 12 56 13 14 4 14 13 56 21 20 21 34 35 44 43 33 43 44 50 55 50 49 48 49 54 0	
Ruta final vehículo 1	1 5 6 15 23 30 28 29 27 22 27 37 36 45 46 51 46 47 38 47 52 0	
	Coste de la ruta	71.9
	Distancia recorrida	75
Ruta final vehículo 2	2 7 17 18 17 16 17 7 9 8 9 24 25 24 26 20 26 32 31 40 41 42 39 53 0	
	Coste de la ruta	83.8
	Distancia recorrida	89
Ruta final vehículo 3	3 12 11 10 11 19 11 12 56 21 34 35 14 13 14 4 14 35 44 43 33 43 44 50 49 48 49 54 49 50 55 0	
	Coste de la ruta	125.7
	Distancia recorrida	130
Coste total de las rutas	281.4	
Distancia total	294	
Tiempo (s)	0.019702	

Tabla 6: Pruebas del fichero de entrada ejemplo con diez iteraciones.

Nº de iteraciones	100	
Ruta vehículo 1	1 5 6 15 23 30 38 47 46 45 36 37 27 22 27 29 28 29 27 37 36 45 46 51 0	
Ruta vehículo 2	2 7 17 18 17 16 17 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
Ruta vehículo 3	3 12 56 21 20 21 34 35 14 4 14 13 14 35 44 43 33 43 44 50 49 48 49 54 0	
Puntos sin visitar	10 11 19 52 55	
Ruta vehículo 1	1 5 6 15 23 30 38 47 52 47 46 45 36 37 27 22 27 29 28 29 27 37 36 45 46 51 0	
Ruta vehículo 2	2 7 17 18 17 16 17 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
Ruta vehículo 3	3 12 11 19 11 10 11 19 11 12 56 21 20 21 34 35 14 4 14 13 14 35 44 43 33 43 44 50 55 50 49 48 49 54 0	
Ruta final vehículo 1	1 5 6 15 23 30 38 47 52 47 46 45 36 37 27 22 27 29 28 29 27 37 36 45 46 51 0	
	Coste de la ruta	90
	Distancia recorrida	93
Ruta final vehículo 2	2 7 17 18 17 16 17 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
	Coste de la ruta	79.8
	Distancia recorrida	85
Ruta final vehículo 3	3 12 11 19 11 10 11 12 56 21 20 21 34 35 14 4 14 13 14 35 44 43 33 43 44 50 55 50 49 48 49 54 0	
	Coste de la ruta	122
	Distancia recorrida	126
Coste total de las rutas	291.8	
Distancia total	304	
Tiempo (s)	0.14085	

Tabla 7: Pruebas del fichero de entrada ejemplo con cien iteraciones.

Nº de iteraciones	1000	
Ruta vehículo 1	1 5 6 15 23 30 38 37 27 22 27 29 28 29 27 37 36 45 46 47 52 0	
Ruta vehículo 2	2 7 17 18 17 16 17 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
Ruta vehículo 3	3 12 11 19 11 10 11 12 56 21 34 35 14 13 14 4 14 35 44 43 33 43 44 50 49 48 49 54 0	
Puntos sin visitar	20 51 55	
Ruta vehículo 1	1 5 6 15 23 30 38 37 27 22 27 29 28 29 27 37 36 45 46 51 46 47 52 0	
Ruta vehículo 2	2 7 17 18 17 16 17 7 9 8 9 24 25 24 26 20 26 32 31 40 41 42 39 53 0	
Ruta vehículo 3	3 12 11 19 11 10 11 12 56 21 34 35 14 13 14 4 14 35 44 43 33 43 44 50 55 50 49 48 49 54 0	
Ruta final vehículo 1	1 5 6 15 23 30 38 37 27 22 27 29 28 29 27 37 36 45 46 51 46 47 52 0	
	Coste de la ruta	81.6
	Distancia recorrida	85
Ruta final vehículo 2	2 7 17 18 17 16 17 7 9 8 9 24 25 24 26 20 26 32 31 40 41 42 39 53 0	
	Coste de la ruta	83.8
	Distancia recorrida	89
Ruta final vehículo 3	3 12 11 19 11 10 11 12 56 21 34 35 14 13 14 4 14 35 44 43 33 43 44 50 55 50 49 48 49 54 0	
	Coste de la ruta	116
	Distancia recorrida	120
Coste total de las rutas	281.4	
Distancia total	294	
Tiempo (s)	1.18151	

Tabla 8: Pruebas del fichero de entrada ejemplo con mil iteraciones.

Nº de iteraciones	10000	
Ruta vehículo 1	1 5 6 15 23 30 28 29 27 22 27 37 38 47 46 45 36 45 46 51 0	
Ruta vehículo 2	2 7 17 18 17 16 17 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
Ruta vehículo 3	3 12 11 10 11 19 11 12 56 13 14 4 14 35 34 21 20 21 34 35 44 43 33 43 44 50 49 48 49 54 0	
Puntos sin visitar	52 55	
Ruta vehículo 1	1 5 6 15 23 30 28 29 27 22 27 37 38 47 52 47 46 45 36 45 46 51 0	
Ruta vehículo 2	2 7 17 18 17 16 17 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
Ruta vehículo 3	3 12 11 10 11 19 11 12 56 13 14 4 14 35 34 21 20 21 34 35 44 43 33 43 44 50 55 50 49 48 49 54 0	
Ruta final vehículo 1	1 5 6 15 23 30 28 29 27 22 27 37 38 47 52 47 46 45 36 45 46 51 0	
	Coste de la ruta	82.3
	Distancia recorrida	85
Ruta final vehículo 2	2 7 17 18 17 16 17 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
	Coste de la ruta	79.8
	Distancia recorrida	85
Ruta final vehículo 3	3 12 11 10 11 19 11 12 56 13 14 4 14 35 34 21 20 21 34 35 44 43 33 43 44 50 55 50 49 48 49 54 0	
	Coste de la ruta	114.5
	Distancia recorrida	120
Coste total de las rutas	276.6	
Distancia total	290	
Tiempo (s)	12.8593	

Tabla 9: Pruebas del fichero de entrada ejemplo con diez mil iteraciones.

Nº de iteraciones	100000	
Ruta vehículo 1	1 5 6 15 23 30 28 29 27 22 27 37 36 45 46 47 38 47 52 0	
Ruta vehículo 2	2 7 17 16 17 18 17 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
Ruta vehículo 3	3 12 11 10 11 19 11 12 56 13 14 4 14 35 34 21 20 21 34 35 44 43 33 43 44 50 49 48 49 54 0	
Puntos sin visitar	51 55	
Ruta vehículo 1	1 5 6 15 23 30 28 29 27 22 27 37 36 45 46 51 46 47 38 47 52 0	
Ruta vehículo 2	2 7 17 16 17 18 17 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
Ruta vehículo 3	3 12 11 10 11 19 11 12 56 13 14 4 14 35 34 21 20 21 34 35 44 43 33 43 44 50 55 50 49 48 49 54 0	
Ruta final vehículo 1	1 5 6 15 23 30 28 29 27 22 27 37 36 45 46 51 46 47 38 47 52 0	
	Coste de la ruta	71.9
	Distancia recorrida	75
Ruta final vehículo 2	2 7 17 16 17 18 17 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
	Coste de la ruta	79.8
	Distancia recorrida	85
Ruta final vehículo 3	3 12 11 10 11 19 11 12 56 13 14 4 14 35 34 21 20 21 34 35 44 43 33 43 44 50 55 50 49 48 49 54 0	
	Coste de la ruta	114.5
	Distancia recorrida	120
Coste total de las rutas	266.2	
Distancia total	280	
Tiempo (s)	127.353	

Tabla 10: Pruebas del fichero de entrada ejemplo con cien mil iteraciones.

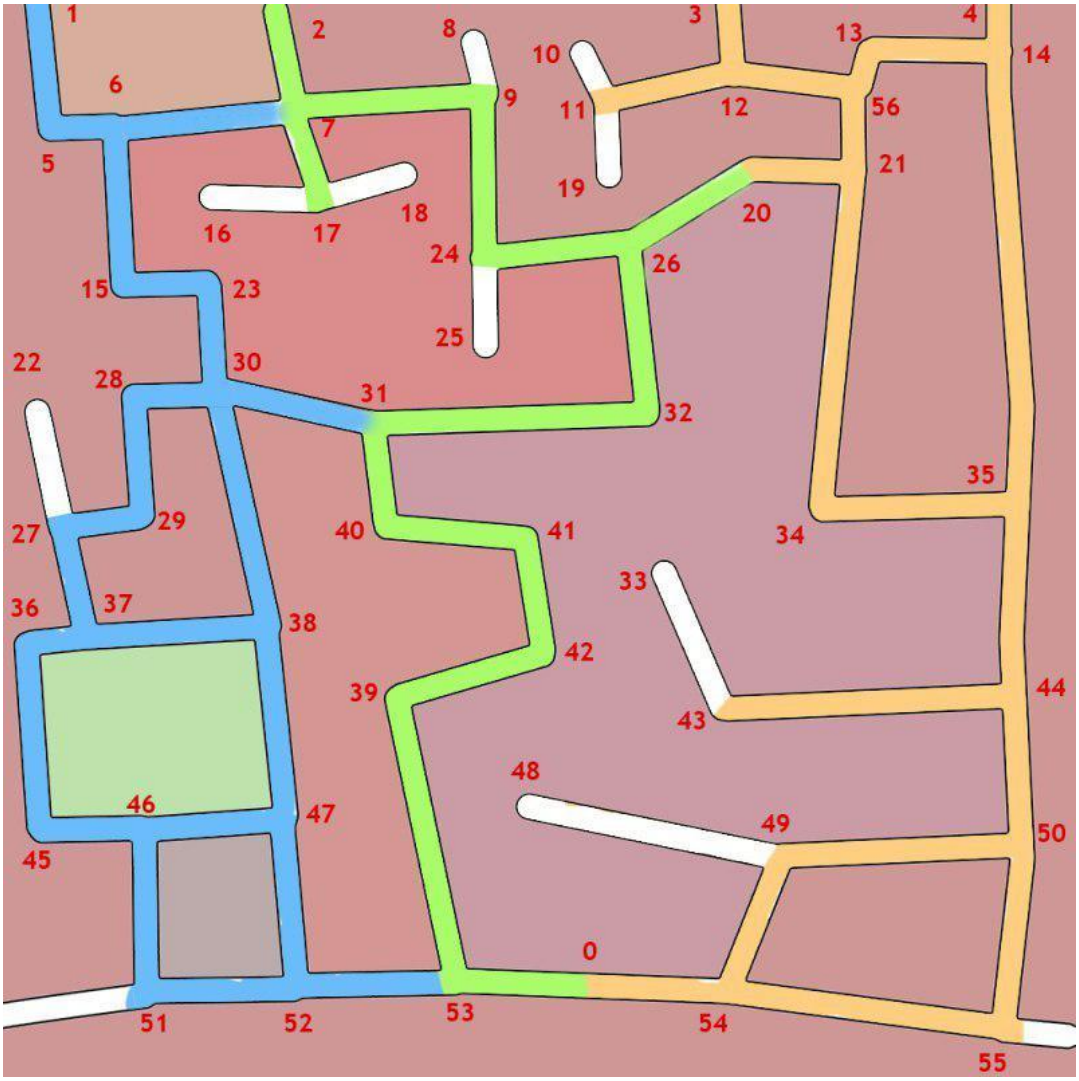

Figura 10: Mapa del municipio con el recorrido de cada vehículo indicado.

GRASP sin estrategias de inclinación

El cálculo del coste de este problema incluye la inclinación de las calles por ser la característica más distintiva de este problema. En cambio, si se quisiera aplicar este algoritmo a lugares con una orografía más simple, tan sólo habría que utilizar la distancia para medir el coste.

Esta tabla representa el coste de las rutas teniendo en cuenta la distancia como coste para las rutas. También se incluye una gráfica para ilustrar la evolución del coste en función de las iteraciones. Cada uno de los puntos representa la distancia que recorren realmente los vehículos.

Nº iteraciones	Coste	Tiempo de ejecución
1	304	0.006957
5	306	0.018252
10	306	0.026995
50	300	0.067834
100	290	0.137
500	304	0.623642
1000	304	1.20705
5000	296	6.24899
10000	304	12.6299
50000	290	62.7447
100000	280	127.086

Tabla 11: Resultados en función del número de iteraciones.

Como se puede observar en la figura número 11, la evolución del coste en función del número de iteraciones es similar a la que se puede observar en la figura número 7. La única diferencia observable es que los costes en esta segunda figura son levemente inferiores.

La figura 12 representa una curva muy similar a la que se puede ver en la figura 8. Es decir, el tiempo de ejecución se incrementa exponenciamente según va incrementando el número de iteraciones.

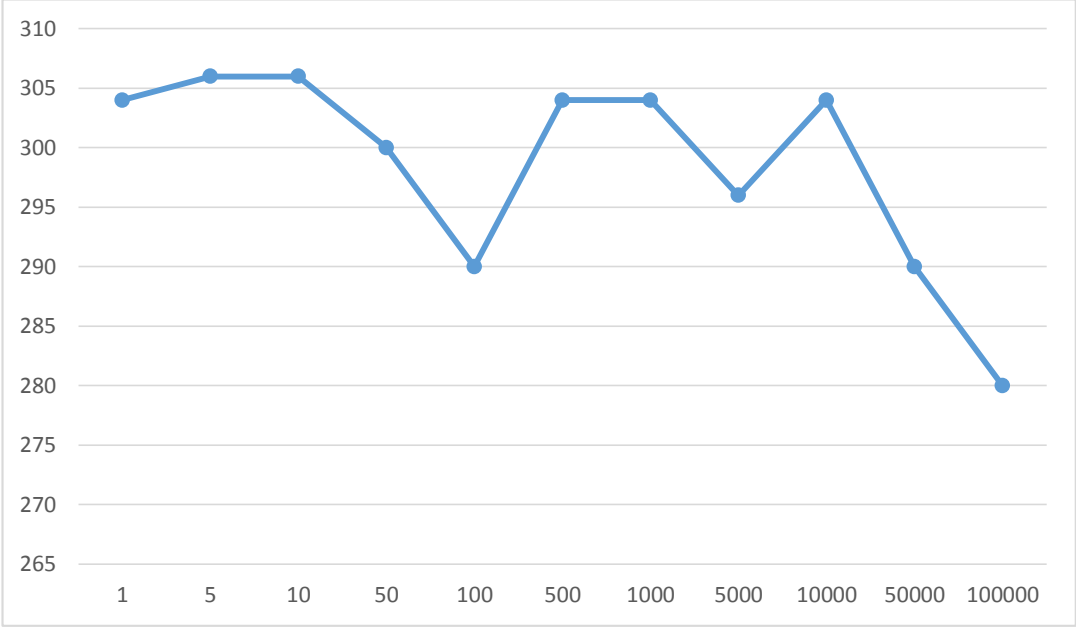

Figura 11: Gráfica de la distancia en función del número de iteraciones.

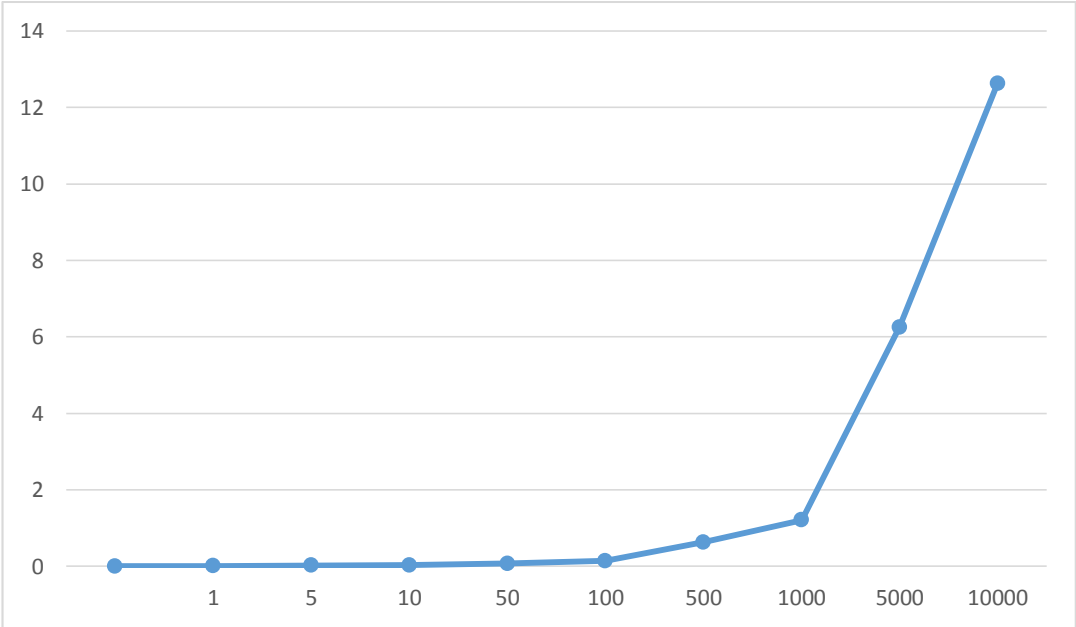

Figura 12: Gráfica del tiempo de ejecución en función del número de iteraciones.

Nº de iteraciones	1	
Ruta vehículo 1	1 5 6 15 23 30 38 47 52 0	
Ruta vehículo 2	2 7 17 16 17 18 17 7 9 8 9 24 26 32 31 40 41 42 39 53 0	
Ruta vehículo 3	3 12 11 10 11 19 11 12 56 21 34 35 14 13 14 4 14 35 44 43 33 43 44 50 49 54 0	
Puntos sin visitar	20 22 25 27 28 29 36 37 45 46 48 51 55	
Ruta vehículo 1	1 5 6 15 23 30 28 29 28 30 38 37 36 45 36 37 27 22 27 37 36 45 36 37 38 47 46 51 46 47 52 0	
Ruta vehículo 2	2 7 17 16 17 18 17 7 9 8 9 24 25 24 26 20 26 32 31 40 41 42 39 53 0	
Ruta vehículo 3	3 12 11 10 11 19 11 12 56 21 34 35 14 13 14 4 14 35 44 43 33 43 44 50 55 50 49 48 49 54 0	
Ruta final vehículo 1	1 5 6 15 23 30 28 29 28 30 38 37 36 45 36 37 27 22 27 37 38 47 46 51 46 47 52 0	
	Distancia recorrida	95
Ruta final vehículo 2	2 7 17 16 17 18 17 7 9 8 9 24 25 24 26 20 26 32 31 40 41 42 39 53 0	
	Distancia recorrida	89
Ruta final vehículo 3	3 12 11 10 11 19 11 12 56 21 34 35 14 13 14 4 14 35 44 43 33 43 44 50 55 50 49 48 49 54 0	
	Distancia recorrida	120
Distancia total	304	
Tiempo (s)	0.006957	

Tabla 12: Pruebas del fichero de entrada ejemplo con una iteración.

Nº de iteraciones	10	
Ruta vehículo 1	1 5 6 15 23 30 38 37 27 22 27 29 28 29 27 37 36 45 46 51 0	
Ruta vehículo 2	2 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
Ruta vehículo 3	3 12 11 10 11 19 11 12 56 13 14 35 34 21 20 21 34 35 44 43 33 43 44 50 55 0	
Puntos sin visitar	4 16 17 18 47 48 49 52 54	
Ruta vehículo 1	1 5 6 15 23 30 38 47 52 47 38 37 27 22 27 29 28 29 27 37 36 45 46 51 0	
Ruta vehículo 2	2 7 17 18 17 16 17 18 17 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
Ruta vehículo 3	3 12 11 10 11 19 11 12 56 13 14 4 14 35 34 21 20 21 34 35 44 43 33 43 44 50 49 48 49 54 49 48 49 50 55 0	
Ruta final vehículo 1	1 5 6 15 23 30 38 47 52 47 38 37 27 22 27 29 28 29 27 37 36 45 46 51 0	
	Distancia recorrida	91
Ruta final vehículo 2	2 7 17 18 17 16 17 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
	Distancia recorrida	85
Ruta final vehículo 3	3 12 11 10 11 19 11 12 56 13 14 4 14 35 34 21 20 21 34 35 44 43 33 43 44 50 49 48 49 54 49 50 55 0	
	Distancia recorrida	130
Distancia total	306	
Tiempo (s)	0.026995	

Tabla 13: Pruebas del fichero de entrada ejemplo con diez iteraciones.

Nº de iteraciones	100	
Ruta vehículo 1	1 5 6 15 23 30 28 29 27 22 27 37 38 47 46 51 0	
Ruta vehículo 2	2 7 17 16 17 18 17 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
Ruta vehículo 3	3 12 11 10 11 19 11 12 56 21 34 35 14 13 14 4 14 35 44 43 33 43 44 50 49 48 49 54 0	
Puntos sin visitar	20 36 45 52 55	
Ruta vehículo 1	1 5 6 15 23 30 28 29 27 22 27 37 36 37 38 47 52 47 46 45 46 51 0	
Ruta vehículo 2	2 7 17 16 17 18 17 7 9 8 9 24 25 24 26 20 26 32 31 40 41 42 39 53 0	
Ruta vehículo 3	3 12 11 10 11 19 11 12 56 21 34 35 14 13 14 4 14 35 44 43 33 43 44 50 55 50 49 48 49 54 0	
Ruta final vehículo 1	1 5 6 15 23 30 28 29 27 22 27 37 36 37 38 47 52 47 46 45 46 51 0	
	Distancia recorrida	81
Ruta final vehículo 2	2 7 17 16 17 18 17 7 9 8 9 24 25 24 26 20 26 32 31 40 41 42 39 53 0	
	Distancia recorrida	89
Ruta final vehículo 3	3 12 11 10 11 19 11 12 56 21 34 35 14 13 14 4 14 35 44 43 33 43 44 50 55 50 49 48 49 54 0	
	Distancia recorrida	120
Distancia total	290	
Tiempo (s)	0.137	

Tabla 14: Pruebas del fichero de entrada ejemplo con cien iteraciones.

Nº de iteraciones	1000	
Ruta vehículo 1	1 5 6 15 23 30 38 47 46 45 36 37 27 22 27 29 28 29 27 37 36 45 46 51 0	
Ruta vehículo 2	2 7 17 16 17 18 17 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
Ruta vehículo 3	3 12 11 19 11 10 11 12 56 21 20 21 34 35 14 4 14 13 14 35 44 43 33 43 44 50 49 48 49 54 0	
Puntos sin visitar	52 55	
Ruta vehículo 1	1 5 6 15 23 30 38 47 52 47 46 45 36 37 27 22 27 29 28 29 27 37 36 45 46 51 0	
Ruta vehículo 2	2 7 17 16 17 18 17 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
Ruta vehículo 3	3 12 11 19 11 10 11 12 56 21 20 21 34 35 14 4 14 13 14 35 44 43 33 43 44 50 55 50 49 48 49 54 0	
Ruta final vehículo 1	1 5 6 15 23 30 38 47 52 47 46 45 36 37 27 22 27 29 28 29 27 37 36 45 46 51 0	
	Distancia recorrida	93
Ruta final vehículo 2	2 7 17 16 17 18 17 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
	Distancia recorrida	85
Ruta final vehículo 3	3 12 11 19 11 10 11 12 56 21 20 21 34 35 14 4 14 13 14 35 44 43 33 43 44 50 55 50 49 48 49 54 0	
	Distancia recorrida	126
Distancia total	304	
Tiempo (s)	1.20705	

Tabla 15: Pruebas del fichero de entrada ejemplo con mil iteraciones.

Nº de iteraciones	10000	
Ruta vehículo 1	1 5 6 15 23 30 38 47 46 45 36 37 27 22 27 29 28 29 27 37 36 45 46 51 0	
Ruta vehículo 2	2 7 17 16 17 18 17 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
Ruta vehículo 3	3 12 11 19 11 10 11 12 56 21 20 21 34 35 14 4 14 13 14 35 44 43 33 43 44 50 49 48 49 54 0	
Puntos sin visitar	52 55	
Ruta vehículo 1	1 5 6 15 23 30 38 47 52 47 46 45 36 37 27 22 27 29 28 29 27 37 36 45 46 51 0	
Ruta vehículo 2	2 7 17 16 17 18 17 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
Ruta vehículo 3	3 12 11 19 11 10 11 12 56 21 20 21 34 35 14 4 14 13 14 35 44 43 33 43 44 50 55 50 49 48 49 54 0	
Ruta final vehículo 1	1 5 6 15 23 30 38 47 52 47 46 45 36 37 27 22 27 29 28 29 27 37 36 45 46 51 0	
	Distancia recorrida	93
Ruta final vehículo 2	2 7 17 16 17 18 17 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
	Distancia recorrida	85
Ruta final vehículo 3	3 12 11 19 11 10 11 12 56 21 20 21 34 35 14 4 14 13 14 35 44 43 33 43 44 50 55 50 49 48 49 54 0	
	Distancia recorrida	126
Distancia total	304	
Tiempo (s)	12.6299	

Tabla 16: Pruebas del fichero de entrada ejemplo con diez mil iteraciones.

Nº de iteraciones	100000	
Ruta vehículo 1	1 5 6 15 23 30 28 29 27 22 27 37 36 45 46 47 38 47 52 0	
Ruta vehículo 2	2 7 17 18 17 16 17 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
Ruta vehículo 3	3 12 11 10 11 19 11 12 56 13 14 4 14 35 34 21 20 21 34 35 44 43 33 43 44 50 49 48 49 54 0	
Puntos sin visitar	51 55	
Ruta vehículo 1	1 5 6 15 23 30 28 29 27 22 27 37 36 45 46 51 46 47 38 47 52 0	
Ruta vehículo 2	2 7 17 18 17 16 17 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
Ruta vehículo 3	3 12 11 10 11 19 11 12 56 13 14 4 14 35 34 21 20 21 34 35 44 43 33 43 44 50 55 50 49 48 49 54 0	
Ruta final vehículo 1	1 5 6 15 23 30 28 29 27 22 27 37 36 45 46 51 46 47 38 47 52 0	
	Distancia recorrida	75
Ruta final vehículo 2	2 7 17 18 17 16 17 7 9 8 9 24 25 24 26 32 31 40 41 42 39 53 0	
	Distancia recorrida	85
Ruta final vehículo 3	3 12 11 10 11 19 11 12 56 13 14 4 14 35 34 21 20 21 34 35 44 43 33 43 44 50 55 50 49 48 49 54 0	
	Distancia recorrida	120
Distancia total	280	
Tiempo (s)	127.086	

Tabla 17: Pruebas del fichero de entrada ejemplo con cien mil iteraciones.

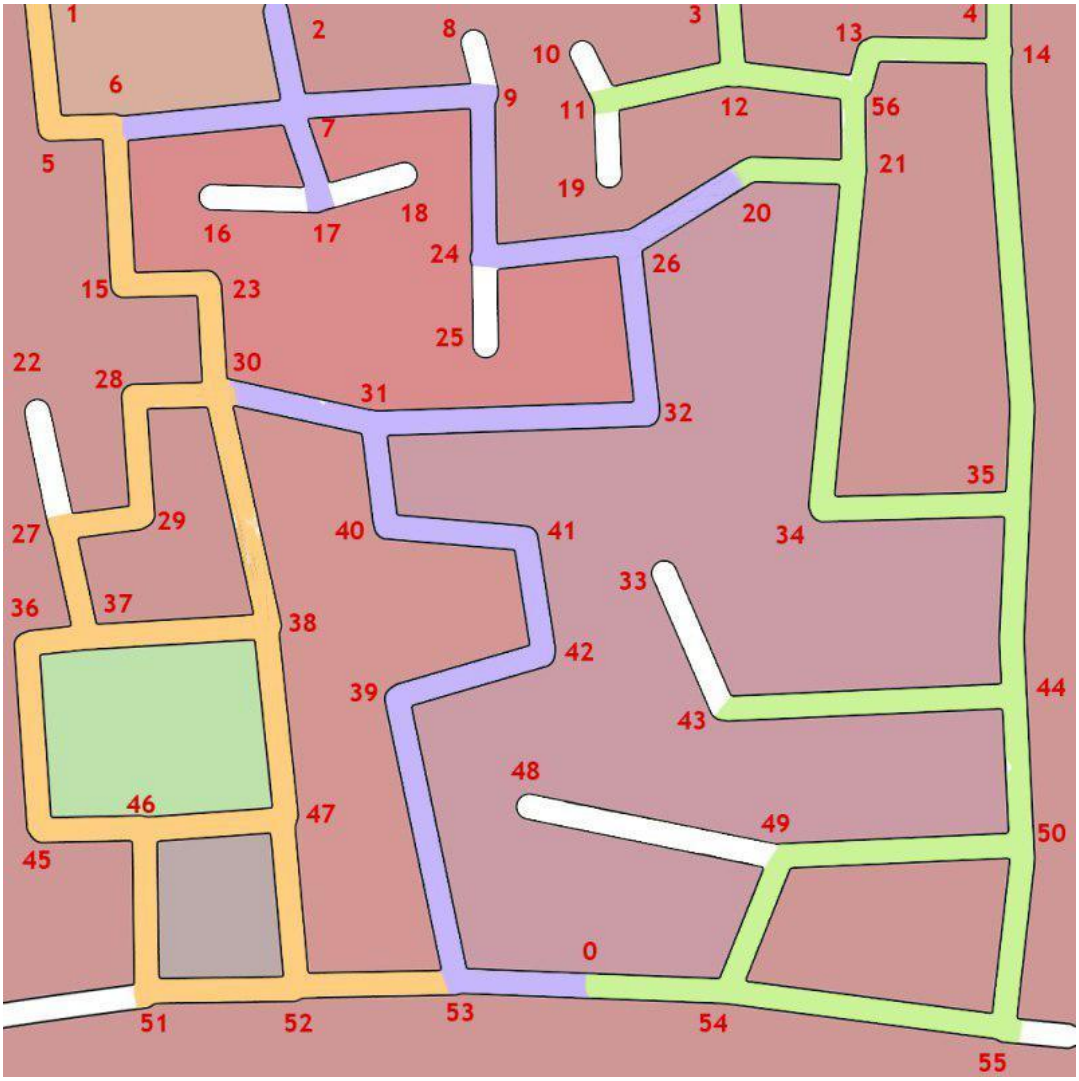

Figura 13: Mapa del municipio con el recorrido de cada vehículo indicado en función de la distancia.

También se ha probado el programa haciendo que los vehículos comiencen sus rutas desde el depósito o punto 0. El problema que surgía era que no se podían encontrar soluciones a causa de las adyacencias, ya que cuando se generaba la ruta del primer y segundo vehículo, prácticamente visitaban todos los puntos adyacentes al origen, y el tercer vehículo no podía avanzar en la ruta, por lo que quedaban muchos puntos sin visitar. Para solucionar este problema, se ha modificado el procedimiento que genera las rutas, de forma que vaya generando las rutas de los tres vehículos en paralelo. Estos son los resultados obtenidos:

Nº iteraciones	**Coste**	**Tiempo de ejecución**
1	330	0.005247
5	382	0.005173
10	391	0.005509
50	384	0.02564
100	342	0.045768
500	337	0.154761
1000	325	0.296319
5000	323	1.42054
10000	353	2.90165
50000	338	14.4718
100000	307	28.6492

Tabla 18: Resultados en función del número de iteraciones.

Como se puede observar, los resultados no son muy eficientes con este tipo de búsqueda de rutas, y tampoco mejoran en gran medida con el incremento en el número de iteraciones. Un dato a destacar acerca de estos resultados es la eficacia del procedimiento de reducción de las rutas en el último paso. Esto sucede porque las rutas generadas forman muchos bucles innecesarios en su búsqueda por encontrar una ruta en el primer paso.

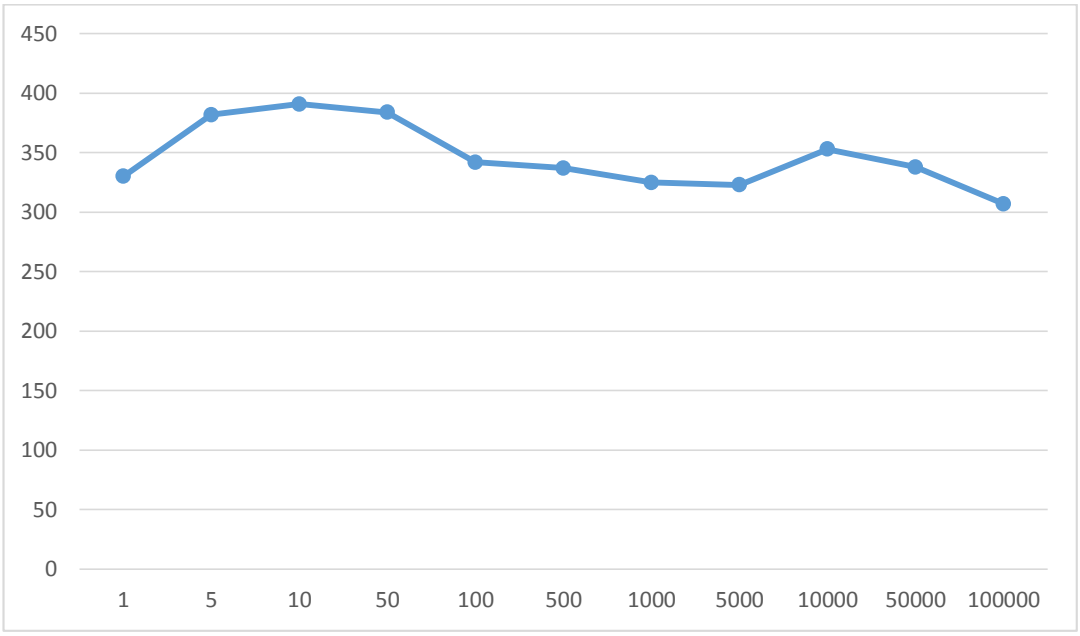

Figura 14: Gráfica de la distancia en función del número de iteraciones.

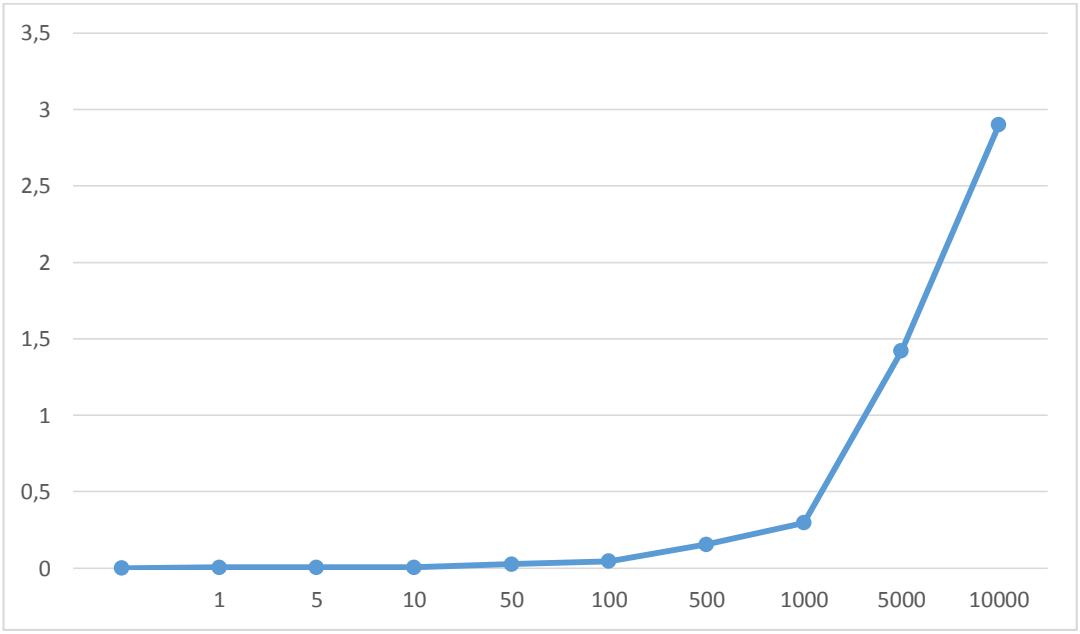

Figura 15: Gráfica del tiempo de ejecución en función del número de iteraciones.

Nº de iteraciones	1	
Ruta vehículo 1	0 55 50 44 43 33 43 44 35 34 21 56 12 11 10 11 19 11 12 3 12 11 10 11 12 56 13 14 4 14 13 56 12 11 12 11 10 11 12 56 21 20	
Ruta vehículo 2	0 53 39 42 41 40 31 32 26 24 25 24 9 8 9 7 17 18 17 16 17 7 6 15 6 5	
Ruta vehículo 3	0 52 47 38 30 28 29 27 37 36 45 46 51 0 54 49 48 49 54 0 51 46 45 36 37 27 22 27 37 36 45 46 51 0 54 49 54 0 51 46 45 36 37 27 29 28 30 23	
Puntos sin visitar	1 2	
Ruta vehículo 1	0 55 50 44 43 33 43 44 35 34 21 56 12 11 10 11 19 11 12 3 12 11 10 11 12 56 13 14 4 14 13 56 12 11 12 11 10 11 12 56 21 20	
Ruta vehículo 2	0 53 39 42 41 40 31 32 26 24 25 24 9 8 9 7 2 7 17 18 17 16 17 7 2 7 6 15 6 5 1	
Ruta vehículo 3	0 52 47 38 30 28 29 27 37 36 45 46 51 0 54 49 48 49 54 0 51 46 45 36 37 27 22 27 37 36 45 46 51 0 54 49 54 0 51 46 45 36 37 27 29 28 30 23	
Ruta final vehículo 1	0 55 50 44 43 33 43 44 35 34 21 56 12 11 10 11 19 11 12 3 12 56 13 14 4 14 13 56 21 20	
	Distancia recorrida	95
Ruta final vehículo 2	0 53 39 42 41 40 31 32 26 24 25 24 9 8 9 7 2 7 17 18 17 16 17 7 6 15 6 5 1 5	
	Distancia recorrida	113
Ruta final vehículo 3	0 52 47 38 30 28 29 27 37 36 45 46 51 0 54 49 48 49 54 0 51 46 45 36 37 27 22 27 29 28 30 23	
	Distancia recorrida	122
Distancia total	330	
Tiempo (s)	0.005247	

Tabla 19: Pruebas del fichero de entrada ejemplo con una iteración.

Nº de iteraciones	10
Ruta vehículo 1	0 55 50 44 43 33 43 44 35 14 4 14 13 56 12 3 12 11 19 11 10 11 12 56 21 20 21 34 21 56 12 11 19 11 12 56 13 14 35 44 43 44 50 49
Ruta vehículo 2	0 52 47 38 37 36 45 46 45 36 37 27 22 27 29 28 30 23 15 6 5 1 5 6 15 23 30 28 29 27 37 36 45 36 37 38 47 52 0 54
Ruta vehículo 3	0 51 0 53 39 42 41 40 31 32 26 24 25 24 9 7 2 7 17 16 17 18 17 7 9 8
Puntos sin visitar	48
Ruta vehículo 1	0 55 50 44 43 33 43 44 35 14 4 14 13 56 12 3 12 11 19 11 10 11 12 56 21 20 21 34 21 56 12 11 19 11 12 56 13 14 35 44 43 44 50 49 48 49
Ruta vehículo 2	0 52 47 38 37 36 45 46 45 36 37 27 22 27 29 28 30 23 15 6 5 1 5 6 15 23 30 28 29 27 37 36 45 36 37 38 47 52 0 54
Ruta vehículo 3	0 51 0 53 39 42 41 40 31 32 26 24 25 24 9 7 2 7 17 16 17 18 17 7 9 8
Ruta final vehículo 1	0 55 50 44 43 33 43 44 35 14 4 14 13 56 12 3 12 11 19 11 10 11 12 56 21 20 21 34 21 56 13 14 35 44 50 49 48 49
	Distancia recorrida 148
Ruta final vehículo 2	0 52 47 38 37 36 45 46 45 36 37 27 22 27 29 28 30 23 15 6 5 1 5 6 15 23 30 28 29 27 37 38 47 52 0 54
	Distancia recorrida 125
Ruta final vehículo 3	0 51 0 53 39 42 41 40 31 32 26 24 25 24 9 7 2 7 17 16 17 18 17 7 9 8
	Distancia recorrida 118
Distancia total	391
Tiempo (s)	0.005509

Tabla 20: Pruebas del fichero de entrada ejemplo con diez iteraciones.

Nº de iteraciones	100
Ruta vehículo 1	0 55 0 53 39 42 41 40 31 32 26 24 25 24 9 8 9 7 2 7 17 18 17 16
Ruta vehículo 2	0 54 49 48 49 50 44 35 14 4 14 13 56 12 11 19 11 10 11 12 3 12 11 19 11 12 56 21 34 21 20 21 56 12 11 12 11 19 11 12 56 13 14 35 44 43
Ruta vehículo 3	0 52 47 46 45 36 37 38 30 28 29 27 22 27 29 28 30 23 15 6 5 1 5 6 15 23 30 28 29 27 29 28 30 38 37 36 45 46 51
Puntos sin visitar	33
Ruta vehículo 1	0 55 0 53 39 42 41 40 31 32 26 24 25 24 9 8 9 7 2 7 17 18 17 16
Ruta vehículo 2	0 54 49 48 49 50 44 35 14 4 14 13 56 12 11 19 11 10 11 12 3 12 11 19 11 12 56 21 34 21 20 21 56 12 11 12 11 19 11 12 56 13 14 35 44 43 33 43
Ruta vehículo 3	0 52 47 46 45 36 37 38 30 28 29 27 22 27 29 28 30 23 15 6 5 1 5 6 15 23 30 28 29 27 29 28 30 38 37 36 45 46 51
Ruta final vehículo 1	0 55 0 53 39 42 41 40 31 32 26 24 25 24 9 8 9 7 2 7 17 18 17 16
	Distancia recorrida 95
Ruta final vehículo 2	0 54 49 48 49 50 44 35 14 4 14 13 56 12 11 19 11 10 11 12 3 12 56 21 34 21 20 21 56 13 14 35 44 43 33 43
	Distancia recorrida 132
Ruta final vehículo 3	0 52 47 46 45 36 37 38 30 28 29 27 22 27 29 28 30 23 15 6 5 1 5 6 15 23 30 38 37 36 45 46 51
	Distancia recorrida 115
Distancia total	342
Tiempo (s)	0.045768

Tabla 21: Pruebas del fichero de entrada ejemplo con cien iteraciones.

Nº de iteraciones	1000	
Ruta vehículo 1	0 53 39 42 41 40 31 32 26 20 21 34 21 20 26 24 9 8 9 7 2 7 17 16 17 18	
Ruta vehículo 2	0 52 0 51 46 47 38 37 36 45 36 37 27 22 27 29 28 30 23 15 6 5 1	
Ruta vehículo 3	0 55 50 44 35 14 13 56 12 11 19 11 10 11 12 3 12 11 19 11 12 56 13 14 4 14 13 56 12 11 12 11 19 11 12 56 13 14 35 44 43 33 43 44 35 14 13 56 12 11 12 56 13 14 13 56 12 11 12 11 19 11 12 56 13 14 35 44 50 49 48 49 54	
Puntos sin visitar	25	
Ruta vehículo 1	0 53 39 42 41 40 31 32 26 20 21 34 21 20 26 24 25 24 9 8 9 7 2 7 17 16 17	
Ruta vehículo 2	0 52 0 51 46 47 38 37 36 45 36 37 27 22 27 29 28 30 23 15 6 5 1	
Ruta vehículo 3	0 55 50 44 35 14 13 56 12 11 19 11 10 11 12 3 12 11 19 11 12 56 13 14 4 14 13 56 12 11 12 11 19 11 12 56 13 14 35 44 43 33 43 44 35 14 13 56 12 11 12 56 13 14 13 56 12 11 12 11 19 11 12 56 13 14 35 44 50 49 48 49 54	
Ruta final vehículo 1	0 53 39 42 41 40 31 32 26 20 21 34 21 20 26 24 25 24 9 8 9 7 2 7 17 16 17 18	
	Distancia recorrida	107
Ruta final vehículo 2	0 52 0 51 46 47 38 37 36 45 36 37 27 22 27 29 28 30 23 15 6 5 1	
	Distancia recorrida	93
Ruta final vehículo 3	3 12 11 19 11 10 11 12 56 21 20 21 34 35 14 4 14 13 14 35 44 43 33 43 44 50 55 50 49 48 49 54 0	
	Distancia recorrida	125
Distancia total	325	
Tiempo (s)	0.296319	

Tabla 22: Pruebas del fichero de entrada ejemplo con mil iteraciones.

Nº de iteraciones	10000
Ruta vehículo 1	0 55 0 52 47 38 37 27 22 27 29 28 30 23 15 6 5 1 5 6 15 23 30 28 29 27 37 36 45 46
Ruta vehículo 2	0 53 39 42 41 40 31 32 26 20 21 34 21 20 26 24 9 8 9 7 2 7 17 18 17 16
Ruta vehículo 3	0 54 49 50 44 43 33 43 44 35 14 13 56 12 3 12 11 19 11 10 11 12 56 13 14 4 14 13 56 12 11 19 11 12 56 13 14 35 44 43 44 50 49 48 49 50 44 35 14 13 56 12 11 12 56 13 14 13 56 12 11 19 11 12 56 13 14 35 44 43 44 50 49 54 0 51
Puntos sin visitar	25
Ruta vehículo 1	0 55 0 52 47 38 37 27 22 27 29 28 30 23 15 6 5 1 5 6 15 23 30 28 29 27 37 36 45 46
Ruta vehículo 2	0 53 39 42 41 40 31 32 26 20 21 34 21 20 26 24 25 24 9 8 9 7 2 7 17 18 17 16
Ruta vehículo 3	0 54 49 50 44 43 33 43 44 35 14 13 56 12 3 12 11 19 11 10 11 12 56 13 14 4 14 13 56 12 11 19 11 12 56 13 14 35 44 43 44 50 49 48 49 50 44 35 14 13 56 12 11 12 56 13 14 13 56 12 11 19 11 12 56 13 14 35 44 43 44 50 49 54 0 51
Ruta final vehículo 1	0 55 0 52 47 38 37 27 22 27 29 28 30 23 15 6 5 1 5 6 15 23 30 28 29 27 37 36 45 46
	Distancia recorrida 107
Ruta final vehículo 2	0 53 39 42 41 40 31 32 26 20 21 34 21 20 26 24 25 24 9 8 9 7 2 7 17 18 17 16
	Distancia recorrida 107
Ruta final vehículo 3	0 54 49 50 44 43 33 43 44 35 14 13 56 12 3 12 11 19 11 10 11 12 56 13 14 4 14 35 44 50 49 48 49 54 0 51
	Distancia recorrida 139
Distancia total	353
Tiempo (s)	2.90165

Tabla 23: Pruebas del fichero de entrada ejemplo con diez mil iteraciones.

Nº de iteraciones	100000	
Ruta vehículo 1	0 55 0 53 39 42 41 40 31 32 26 24 25 24 9 7 17 16 17 18 17 7 2 7 17 16 17 7 9 8	
Ruta vehículo 2	0 54 49 48 49 50 44 35 34 21 56 12 11 19 11 10 11 12 3 12 11 19 11 12 56 13 14 4 14 13 56 12 11 12 11 19 11 12 56 21 20 21 56 12 11 12 56 13 14 13 56 12 11 12 11 19 11 12 56 21 34 35 44 43	
Ruta vehículo 3	0 51 46 45 36 37 27 22 27 29 28 30 38 47 52 47 38 30 23 15 6 5 1	
Puntos sin visitar	33	
Ruta vehículo 1	0 55 0 53 39 42 41 40 31 32 26 24 25 24 9 7 17 16 17 18 17 7 2 7 17 16 17 7 9 8	
Ruta vehículo 2	0 54 49 48 49 50 44 35 34 21 56 12 11 19 11 10 11 12 3 12 11 19 11 12 56 13 14 4 14 13 56 12 11 12 11 19 11 12 56 21 20 21 56 12 11 12 56 13 14 13 56 12 11 12 11 19 11 12 56 21 34 35 44 43 33 43	
Ruta vehículo 3	0 51 46 45 36 37 27 22 27 29 28 30 38 47 52 47 38 30 23 15 6 5 1	
Ruta final vehículo 1	0 55 0 53 39 42 41 40 31 32 26 24 25 24 9 7 17 16 17 18 17 7 2 7 9 8	
	Distancia recorrida	104
Ruta final vehículo 2	0 54 49 48 49 50 44 35 34 21 56 12 11 19 11 10 11 12 3 12 56 13 14 4 14 13 56 21 20 21 34 35 44 43 33 43	
	Distancia recorrida	118
Ruta final vehículo 3	0 51 46 45 36 37 27 22 27 29 28 30 38 47 52 47 38 30 23 15 6 5 1	
	Distancia recorrida	85
Distancia total	307	
Tiempo (s)	28.6492	

Tabla 24: Pruebas del fichero de entrada ejemplo con cien mil iteraciones.

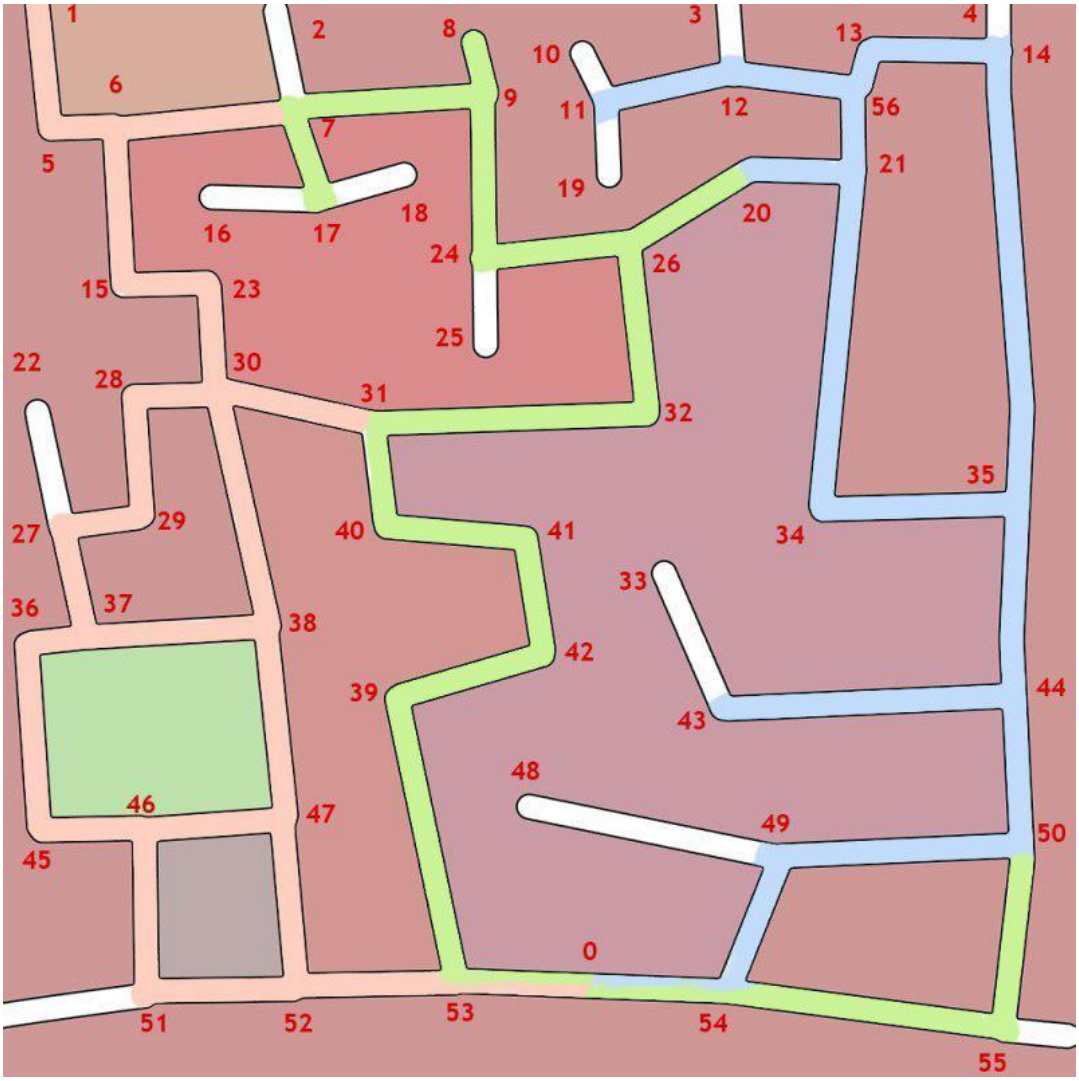

Figura 16: Mapa del municipio con el recorrido de cada vehículo indicado.

CAPÍTULO 8

8. CONCLUSIONES

Tras este largo proceso de investigación, se concluye que es realmente necesario un trato concreto a este tipo de situaciones. Es muy importante que a la hora de implementar la gestión de rutas en zonas rurales se tengan en cuenta todas estas condiciones particulares, ya que de cara a las empresas, este tipo de optimización de las rutas de recogida de residuos supone importantes ahorros no sólo en combustible y desgaste del vehículo, sino también en las horas de trabajo necesarias para gestionar dichas rutas o incluso en el número de vehículos necesarios.

Los resultados obtenidos de la experimentación con el procedimiento propuesto son bastante positivos incluso sin necesidad de un gran número de iteraciones. Además, los tiempos de ejecución no son muy elevados pese a la cantidad de iteraciones a la que es sometido el programa.

En un futuro se podría seguir desarrollando este tema, incluso volver a hacer una toma de contacto con el municipio que dio paso al planteamiento de este proyecto, para por fin ayudarles. De esta forma se podría realizar una comparativa con datos reales de las rutas que actualmente realiza su flota de vehículos, lo cual sería muy enriquecedor.

Como trabajos futuros también sería muy interesante ampliar el programa para abarcar también el Problema de Rutas de Arcos Capacitados con ventanas de tiempo (Capacitated Arc Routing Problem Time Windows o CARP-TW), con instalaciones intermedias (Capacitated Arc Routing Problem with Intermediate Facilities o CARP-IF), con múltiples depósitos (Multiple Depots Capacitated Arc Routing Problem MD-CARP) o incluso ofreciendo la posibilidad de generar rutas en función de un periodo de planificación más adaptado gracias al Problema de Rutas de Arcos Capacitados periódico (Periodic Capacitated Arc Routing Problem o P-CARP).

Personalmente, pienso que aún hay mucho que explotar en el campo de optimización de rutas, no sólo de recogida de residuos sino en todo lo que a gestión de rutas se refiere. La gestión de rutas está en muchos campos de la vida cotidiana, el servicio de envíos postales, las empresas de entrega de mercancía, los servicios de transporte público,… Todos estos sectores se verían enormemente beneficiados.

Llegado el término de este proyecto, me siento muy orgullosa del trabajo realizado, mas me habría gustado haber podido ayudar al municipio gracias al cual surgió todo. A lo largo de todos estos meses este trabajo me ha ayudado no sólo a formarme en un campo que prácticamente desconocía, sino también a adquirir competencias tan valiosas como la autonomía, la competencia para el aprendizaje permanente, para el manejo de la información y, sobre todo, competencia para el manejo de situaciones.

CAPÍTULO 9

9. SUMMARY AND CONCLUSIONS

After this long time of investigation, I think that is really necessary a concrete treatment of that kind of situations. It is really important to implement a complete route management that takes care all those singular conditions. This would generate a great savings for businesses.

The results are quite positive even without a large number of iterations. Despite the large number of iterations to which the program is subjected, execution times are not very high.

In the future there is a lot of things to continue exploring, even retry a contact with the municipality that starts all this to finally help them. That way we could make a good comparison with the actual routes they make.

As future works, would be interesting to expand the program to include also the Capacitated Arc Routing Problem with time windows (CARP-TW), with intermediate facilities (CARP-IF), with multiple deposits (MD-CARP) or even offering the possibility to generate routes based on a more adapted planning period through Periodic Capacitated Arc Routing Problem (P-CARP).

In my opinion, there is much to explore in this area of routing management. Routing management takes part of our lives in many ways: postal services, delivery services, public transport… And all these areas would benefit.

Reached the goal of the project, I feel very proud of the work we have done. Although I would have liked to have been able to help the town thanks to which came all. Throughout all these months this work has helped not only training me in a field practically unkown for me, but also acquire valuable skills such as autonomy, skills for lifelong learning, for the management of information and skills for the management of situations.

CAPÍTULO 10

10. BIBLIOGRAFÍA

[1] Toth, P., & Vigo, D. (Eds.). (2001). *The vehicle routing problem*. Siam.

[2] Thomas Glyn Hinton (2010): "*A Thesis Regarding The Vehicle Routing Problem including a range of Novel Techniques for its Solution*". A dissertation submitted to the **University of Bristol** in accordance with the requirements for award of degree of Doctor of Philosophy in the Faculty of Engineering Department of Computer Science. https://www.cs.bris.ac.uk/~hinton/thesis/thesis.pdf

[3] G. Dantzig, R. Fulkerson, and S. Johnson. Solution of a large-scale traveling-salesman problem. Operations Research, 2:393-410, 1954.

[4] G. B. Dantzig and J. H. Ramser. The truck dispatching problem. Management Science, 6:80-91, 10 1959.

[5] J. S. DeArmon. A Comparison of Heuristics for the Capacitated Chinese Postman Problem. Master's thesis, University of Maryland, Colledge Park, MD, 1981.

[6] M. Dror. Arc Routing: Theory, Solutions and Applications. Kuwer Academic Publishers, 2000.

[7] H. A. Eiselt, M. Gendreau, and G. Laporte. Arc routing problems II: The Rural Postman Problem. Operations Research, 43(3):339-414, 1995.

[8] Maniezzo, V. Algotithms for large directed CARP instances: urban solid waste collection operational support. Technical Report UBLCS-2004-16, University of Bologna, Department of Computer Science, 2004.

[9] Golden, B., and Wong, R. Capacitated arc routing problems. Networks 11 (1981), 47-59

[10] Lacomme, P., Prins, C., & Ramdane-Chérif, W. (2001). A genetic algorithm for the capacitated arc routing problem and its extensions. In *Applications of evolutionary computing* (pp. 473-483). Springer Berlin Heidelberg.

[11] Beullens, P., Muyldermans, L., Cattrysse, D., & Van Oudheusden, D. (2003). A guided local search heuristic for the capacitated arc routing problem. *European Journal of Operational Research*, *147*(3), 629-643.

[12] Brandão, J., & Eglese, R. (2008). A deterministic tabu search algorithm for the capacitated arc routing problem. *Computers & Operations Research*, *35*(4), 1112-1126.

[13] Belenguer, J. M., & Benavent, E. (2003). A cutting plane algorithm for the capacitated arc routing problem. *Computers & Operations Research*, *30*(5), 705-728.

[14] Ghiani, G., Guerriero, F., Improta, G., and Musmanno, R. Waste collection in southern Italy: solution of a real-life arc routing problema. International Transactions in Operational Research 12 (2005), 135-144.

ANEXO I

MATRIZ DE DISTANCIAS (X: 0-56; Y :0-14)

	0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*
0	0	999	999	999	999	999	999	999	999	999	999	999	999	999	999
1	999	0	999	999	999	4.0	999	999	999	999	999	999	999	999	999
2	999	999	0	999	999	999	999	4.0	999	999	999	999	999	999	999
3	999	999	999	0	999	999	999	999	999	999	999	999	4.0	999	999
4	999	999	999	999	0	999	999	999	999	999	999	999	999	999	3.0
5	999	4.0	999	999	999	0	1.0	999	999	999	999	999	999	999	999
6	999	999	999	999	999	1.0	0	5.0	999	999	999	999	999	999	999
7	999	999	4.0	999	999	999	5.0	0	999	5.0	999	999	999	999	999
8	999	999	999	999	999	999	999	999	0	2.0	999	999	999	999	999
9	999	999	999	999	999	999	999	5.0	2.0	0	999	999	999	999	999
10	999	999	999	999	999	999	999	999	999	999	0	1.0	999	999	999
11	999	999	999	999	999	999	999	999	999	999	1.0	0	2.0	999	999
12	999	999	999	4.0	999	999	999	999	999	999	999	2.0	0	999	999
13	999	999	999	999	999	999	999	999	999	999	999	999	999	0	4.0
14	999	999	999	999	3.0	999	999	999	999	999	999	999	999	4.0	0
15	999	999	999	999	999	999	5.0	999	999	999	999	999	999	999	999
16	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
17	999	999	999	999	999	999	999	3.0	999	999	999	999	999	999	999
18	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
19	999	999	999	999	999	999	999	999	999	999	999	2.0	999	999	999
20	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
21	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
22	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
23	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
24	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
25	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
26	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
27	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
28	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
29	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
30	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
31	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
32	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
33	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
34	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
35	999	999	999	999	999	999	999	999	999	999	999	999	999	999	11.0

36	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
37	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
38	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
39	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
40	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
41	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
42	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
43	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
44	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
45	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
46	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
47	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
48	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
49	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
50	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
51	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
52	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
53	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
54	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
55	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
56	999	999	999	999	999	999	999	999	999	999	999	999	3.0	1.0	999

MATRIZ DE DISTANCIAS (X: 0-56; Y: 15-29)

	15	*16*	*17*	*18*	*19*	*20*	*21*	*22*	*23*	*24*	*25*	*26*	*27*	*28*	*29*
0	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
1	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
2	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
3	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
4	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
5	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
6	5.0	999	999	999	999	999	999	999	999	999	999	999	999	999	999
7	999	999	3.0	999	999	999	999	999	999	999	999	999	999	999	999
8	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
9	999	999	999	999	999	999	999	999	999	5.0	999	999	999	999	999
10	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
11	999	999	999	999	2.0	999	999	999	999	999	999	999	999	999	999
12	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
13	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
14	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
15	0	999	999	999	999	999	999	999	5.0	999	999	999	999	999	999
16	999	0	3.0	999	999	999	999	999	999	999	999	999	999	999	999
17	999	3.0	0	3.0	999	999	999	999	999	999	999	999	999	999	999
18	999	999	3.0	0	999	999	999	999	999	999	999	999	999	999	999

19	999	999	999	999	0	999	999	999	999	999	999	999	999	999	999
20	999	999	999	999	999	0	3.0	999	999	999	999	2.0	999	999	999
21	999	999	999	999	999	999	0	999	999	999	999	999	999	999	999
22	999	999	999	999	999	999	999	0	999	999	999	999	4.0	999	999
23	5.0	999	999	999	999	999	999	999	0	999	999	999	999	999	999
24	999	999	999	999	999	999	999	999	999	0	3.0	4.0	999	999	999
25	999	999	999	999	999	999	999	999	999	3.0	0	999	999	999	999
26	999	999	999	999	999	2.0	999	999	999	4.0	999	0	999	999	999
27	999	999	999	999	999	999	999	4.0	999	999	999	999	0	999	3.0
28	999	999	999	999	999	999	999	999	999	999	999	999	999	0	2.0
29	999	999	999	999	999	999	999	999	999	999	999	999	999	2.0	0
30	999	999	999	999	999	999	999	999	4.0	999	999	999	999	2.0	999
31	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
32	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
33	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
34	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
35	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
36	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
37	999	999	999	999	999	999	999	999	999	999	999	999	3.0	999	999
38	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
39	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
40	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
41	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
42	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
43	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
44	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
45	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
46	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
47	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
48	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
49	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
50	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
51	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
52	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
53	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
54	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
55	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
56	999	999	999	999	999	999	3.0	999	999	999	999	999	999	999	999

MATRIZ DE DISTANCIAS (X: 0-56; Y: 30-44)

	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44
0	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
1	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
2	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
3	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
4	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
5	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
6	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
7	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
8	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
9	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
10	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
11	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
12	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
13	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
14	999	999	999	999	999	11.0	999	999	999	999	999	999	999	999	999
15	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
16	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
17	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
18	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
19	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
20	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
21	999	999	999	999	7.0	999	999	999	999	999	999	999	999	999	999
22	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
23	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
24	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
25	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
26	999	999	4.0	999	999	999	999	999	999	999	999	999	999	999	999
27	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
28	2.0	999	999	999	999	999	999	999	999	999	999	999	999	999	999
29	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
30	0	3.0	999	999	999	999	999	999	5.0	999	999	999	999	999	999
31	999	0	6.0	999	999	999	999	999	999	999	3.0	999	999	999	999
32	999	6.0	0	999	999	999	999	999	999	999	999	999	999	999	999
33	999	999	999	0	999	999	999	999	999	999	999	999	999	3.0	999
34	999	999	999	999	0	4.0	999	999	999	999	999	999	999	999	999
35	999	999	999	999	4.0	0	999	999	999	999	999	999	999	999	3.0
36	999	999	999	999	999	999	0	1.0	999	999	999	999	999	999	999
37	999	999	999	999	999	999	1.0	0	5.0	999	999	999	999	999	999
38	5.0	999	999	999	999	999	999	5.0	0	999	999	999	999	999	999
39	999	999	999	999	999	999	999	999	999	0	999	999	5.0	999	999
40	999	3.0	999	999	999	999	999	999	999	999	0	4.0	999	999	999
41	999	999	999	999	999	999	999	999	999	999	4.0	0	3.0	999	999
42	999	999	999	999	999	999	999	999	5.0	999	999	3.0	0	999	999

43	999	999	999	3.0	999	999	999	999	999	999	999	999	999	0	6.0
44	999	999	999	999	999	3.0	999	999	999	999	999	999	999	6.0	0
45	999	999	999	999	999	999	3.0	999	999	999	999	999	999	999	999
46	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
47	999	999	999	999	999	999	999	999	3.0	999	999	999	999	999	999
48	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
49	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
50	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
51	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
52	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
53	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
54	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
55	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
56	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999

MATRIZ DE DISTANCIAS (X: 0-56; Y: 45-56)

	45	*46*	*47*	*48*	*49*	*50*	*51*	*52*	*53*	*54*	*55*	*56*
0	999	999	999	999	999	999	999	999	999	999	999	999
1	999	999	999	999	999	999	999	999	999	999	999	999
2	999	999	999	999	999	999	999	999	999	999	999	999
3	999	999	999	999	999	999	999	999	999	999	999	999
4	999	999	999	999	999	999	999	999	999	999	999	999
5	999	999	999	999	999	999	999	999	999	999	999	999
6	999	999	999	999	999	999	999	999	999	999	999	999
7	999	999	999	999	999	999	999	999	999	999	999	999
8	999	999	999	999	999	999	999	999	999	999	999	999
9	999	999	999	999	999	999	999	999	999	999	999	999
10	999	999	999	999	999	999	999	999	999	999	999	999
11	999	999	999	999	999	999	999	999	999	999	999	999
12	999	999	999	999	999	999	999	999	999	999	999	3.0
13	999	999	999	999	999	999	999	999	999	999	999	1.0
14	999	999	999	999	999	999	999	999	999	999	999	999
15	999	999	999	999	999	999	999	999	999	999	999	999
16	999	999	999	999	999	999	999	999	999	999	999	999
17	999	999	999	999	999	999	999	999	999	999	999	999
18	999	999	999	999	999	999	999	999	999	999	999	999
19	999	999	999	999	999	999	999	999	999	999	999	999
20	999	999	999	999	999	999	999	999	999	999	999	999
21	999	999	999	999	999	999	999	999	999	999	999	3.0
22	999	999	999	999	999	999	999	999	999	999	999	999
23	999	999	999	999	999	999	999	999	999	999	999	999
24	999	999	999	999	999	999	999	999	999	999	999	999
25	999	999	999	999	999	999	999	999	999	999	999	999

26	999	999	999	999	999	999	999	999	999	999	999	999
27	999	999	999	999	999	999	999	999	999	999	999	999
28	999	999	999	999	999	999	999	999	999	999	999	999
29	999	999	999	999	999	999	999	999	999	999	999	999
30	999	999	999	999	999	999	999	999	999	999	999	999
31	999	999	999	999	999	999	999	999	999	999	999	999
32	999	999	999	999	999	999	999	999	999	999	999	999
33	999	999	999	999	999	999	999	999	999	999	999	999
34	999	999	999	999	999	999	999	999	999	999	999	999
35	999	999	999	999	999	999	999	999	999	999	999	999
36	3.0	999	999	999	999	999	999	999	999	999	999	999
37	999	999	999	999	999	999	999	999	999	999	999	999
38	999	999	3.0	999	999	999	999	999	999	999	999	999
39	999	999	999	999	999	999	999	999	6.0	999	999	999
40	999	999	999	999	999	999	999	999	999	999	999	999
41	999	999	999	999	999	999	999	999	999	999	999	999
42	999	999	999	999	999	999	999	999	999	999	999	999
43	999	999	999	999	999	999	999	999	999	999	999	999
44	999	999	999	999	999	999	999	999	999	999	999	999
45	0	999	999	999	999	999	999	999	999	999	999	999
46	999	0	3.0	999	999	999	4.0	999	999	999	999	999
47	999	3.0	0	999	999	999	999	4.0	999	999	999	999
48	999	999	999	0	6.0	999	999	999	999	999	999	999
49	999	999	999	6.0	0	5.0	999	999	999	999	999	999
50	999	999	999	999	5.0	0	999	999	999	999	999	999
51	999	4.0	999	999	999	999	0	999	999	999	999	999
52	999	999	4.0	999	999	999	999	0	999	999	999	999
53	999	999	999	999	999	999	999	999	0	999	999	999
54	999	999	999	999	3.0	999	999	999	999	0	999	999
55	999	999	999	999	999	3.0	999	999	999	999	0	999
56	999	999	999	999	999	999	999	999	999	999	999	0

MATRIZ DE INCLINACIÓN (X: 0-56; Y: 0-14)

	0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*
0	0	999	999	999	999	999	999	999	999	999	999	999	999	999	999
1	999	0	999	999	999	-0.1	999	999	999	999	999	999	999	999	999
2	999	999	0	999	999	999	999	-0.2	999	999	999	999	999	999	999
3	999	999	999	0	999	999	999	999	999	999	999	999	-0.3	999	999
4	999	999	999	999	0	999	999	999	999	999	999	999	999	999	-0.1
5	999	+0.1	999	999	999	0	0	999	999	999	999	999	999	999	999
6	999	999	999	999	999	0	0	0	999	999	999	999	999	999	999
7	999	999	+0.2	999	999	999	0	0	999	0	999	999	999	999	999
8	999	999	999	999	999	999	999	999	0	-0.3	999	999	999	999	999

9	999	999	999	999	999	999	999	0	+0.3	0	999	999	999	999	999
10	999	999	999	999	999	999	999	999	999	999	0	-0.1	999	999	999
11	999	999	999	999	999	999	999	999	999	999	+0.1	0	0	999	999
12	999	999	999	+0.3	999	999	999	999	999	999	999	0	0	999	999
13	999	999	999	999	999	999	999	999	999	999	999	999	999	0	0
14	999	999	999	999	+0.1	999	999	999	999	999	999	999	999	0	0
15	999	999	999	999	999	999	+0.1	999	999	999	999	999	999	999	999
16	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
17	999	999	999	999	999	999	999	-0.2	999	999	999	999	999	999	999
18	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
19	999	999	999	999	999	999	999	999	999	999	999	+0.2	999	999	999
20	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
21	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
22	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
23	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
24	999	999	999	999	999	999	999	999	999	+0.3	999	999	999	999	999
25	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
26	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
27	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
28	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
29	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
30	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
31	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
32	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
33	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
34	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
35	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
36	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
37	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
38	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
39	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
40	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
41	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
42	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
43	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
44	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
45	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
46	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
47	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
48	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
49	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
50	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
51	0	999	999	999	999	999	999	999	999	999	999	999	999	999	999
52	0	999	999	999	999	999	999	999	999	999	999	999	999	999	999
53	0	999	999	999	999	999	999	999	999	999	999	999	999	999	999
54	0	999	999	999	999	999	999	999	999	999	999	999	999	999	999
55	0	999	999	999	999	999	999	999	999	999	999	999	999	999	999

56	999	999	999	999	999	999	999	999	999	999	999	999	0	+0.1	999

MATRIZ DE INCLINACIÓN (X: 0-56; Y: 15-29)

	15	*16*	*17*	*18*	*19*	*20*	*21*	*22*	*23*	*24*	*25*	*26*	*27*	*28*	*29*
0	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
1	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
2	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
3	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
4	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
5	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
6	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
7	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
8	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
9	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
10	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
11	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
12	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
13	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
14	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
15	0	999	999	999	999	999	999	999	0	999	999	999	999	999	999
16	999	0	0	999	999	999	999	999	999	999	999	999	999	999	999
17	999	0	0	0	999	999	999	999	999	999	999	999	999	999	999
18	999	999	0	0	999	999	999	999	999	999	999	999	999	999	999
19	999	999	999	999	0	999	999	999	999	999	999	999	999	999	999
20	999	999	999	999	999	0	0	999	999	999	999	-0.1	999	999	999
21	999	999	999	999	999	0	0	999	999	999	999	999	999	999	999
22	999	999	999	999	999	999	999	0	999	999	999	999	-0.2	999	999
23	0	999	999	999	999	999	999	999	0	999	999	999	999	999	999
24	999	999	999	999	999	999	999	999	999	0	-0.1	0	999	999	999
25	999	999	999	999	999	999	999	999	999	+0.1	0	999	999	999	999
26	999	999	999	999	999	+0.1	999	999	999	0	999	0	999	999	999
27	999	999	999	999	999	999	999	+0.2	999	999	999	999	0	999	0
28	999	999	999	999	999	999	999	999	999	999	999	999	999	0	-0.2
29	999	999	999	999	999	999	999	999	999	999	999	999	0	+0.2	0
30	999	999	999	999	999	999	999	999	+0.1	999	999	999	999	0	999
31	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
32	999	999	999	999	999	999	999	999	999	999	999	+0.2	999	999	999
33	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
34	999	999	999	999	999	999	+0.2	999	999	999	999	999	999	999	999
35	999	999	999	999	999	999	999	999	999	999	999	999	999	999	+0.3
36	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
37	999	999	999	999	999	999	999	999	999	999	999	999	+0.1	999	999
38	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999

39	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
40	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
41	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
42	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
43	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
44	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
45	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
46	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
47	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
48	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
49	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
50	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
51	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
52	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
53	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
54	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
55	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
56	999	999	999	999	999	999	+0.1	999	999	999	999	999	999	999	999

MATRIZ DE INCLINACIÓN (X: 0-56; Y: 30-44)

	30	*31*	*32*	*33*	*34*	*35*	*36*	*37*	*38*	*39*	*40*	*41*	*42*	*43*	*44*
0	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
1	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
2	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
3	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
4	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
5	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
6	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
7	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
8	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
9	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
10	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
11	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
12	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
13	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
14	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
15	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
16	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
17	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
18	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
19	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
20	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
21	999	999	999	999	-0.2	999	999	999	999	999	999	999	999	999	999

22	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
23	-0.1	999	999	999	999	999	999	999	999	999	999	999	999	999	999
24	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
25	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
26	999	999	-0.2	999	999	999	999	999	999	999	999	999	999	999	999
27	999	999	999	999	999	999	999	-0.1	999	999	999	999	999	999	999
28	0	999	999	999	999	999	999	999	999	999	999	999	999	999	999
29	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
30	0	0	999	999	999	999	999	999	-0.2	999	999	999	999	999	999
31	0	0	0	999	999	999	999	999	999	999	-0.1	999	999	999	999
32	999	0	0	999	999	999	999	999	999	999	999	999	999	999	999
33	999	999	999	0	999	999	999	999	999	999	999	999	999	-0.2	999
34	999	999	999	999	0	0	999	999	999	999	999	999	999	999	999
35	999	999	999	999	0	0	999	999	999	999	999	999	999	999	-0.1
36	999	999	999	999	999	999	0	0	999	999	999	999	999	999	999
37	999	999	999	999	999	999	0	0	0	999	999	999	999	999	999
38	+0.2	999	999	999	999	999	999	0	0	999	999	999	999	999	999
39	999	999	999	999	999	999	999	999	999	0	999	999	0	999	999
40	999	+0.1	999	999	999	999	999	999	999	999	0	0	999	999	999
41	999	999	999	999	999	999	999	999	999	999	0	0	-0.2	999	999
42	999	999	999	999	999	999	999	999	999	0	999	+0.2	0	999	999
43	999	999	999	+0.2	999	999	999	999	999	999	999	999	999	0	0
44	999	999	999	999	999	+0.1	999	999	999	999	999	999	999	0	0
45	999	999	999	999	999	999	+0.1	999	999	999	999	999	999	999	999
46	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
47	999	999	999	999	999	999	999	999	+0.1	999	999	999	999	999	999
48	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
49	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
50	999	999	999	999	999	999	999	999	999	999	999	999	999	999	+0.1
51	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
52	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
53	999	999	999	999	999	999	999	999	999	+0.2	999	999	999	999	999
54	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
55	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
56	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999

MATRIZ DE INCLINACIÓN (X: 0-56; Y: 45-56)

	45	*46*	*47*	*48*	*49*	*50*	*51*	*52*	*53*	*54*	*55*	*56*
0	999	999	999	999	999	999	999	999	999	999	999	999
1	999	999	999	999	999	999	999	999	999	999	999	999
2	999	999	999	999	999	999	999	999	999	999	999	999
3	999	999	999	999	999	999	999	999	999	999	999	999
4	999	999	999	999	999	999	999	999	999	999	999	999

5	999	999	999	999	999	999	999	999	999	999	999	999
6	999	999	999	999	999	999	999	999	999	999	999	999
7	999	999	999	999	999	999	999	999	999	999	999	999
8	999	999	999	999	999	999	999	999	999	999	999	999
9	999	999	999	999	999	999	999	999	999	999	999	999
10	999	999	999	999	999	999	999	999	999	999	999	999
11	999	999	999	999	999	999	999	999	999	999	999	999
12	999	999	999	999	999	999	999	999	999	999	999	999
13	999	999	999	999	999	999	999	999	999	999	999	999
14	999	999	999	999	999	999	999	999	999	999	999	999
15	999	999	999	999	999	999	999	999	999	999	999	999
16	999	999	999	999	999	999	999	999	999	999	999	999
17	999	999	999	999	999	999	999	999	999	999	999	999
18	999	999	999	999	999	999	999	999	999	999	999	999
19	999	999	999	999	999	999	999	999	999	999	999	999
20	999	999	999	999	999	999	999	999	999	999	999	999
21	999	999	999	999	999	999	999	999	999	999	999	+0.1
22	999	999	999	999	999	999	999	999	999	999	999	999
23	999	999	999	999	999	999	999	999	999	999	999	999
24	999	999	999	999	999	999	999	999	999	999	999	999
25	999	999	999	999	999	999	999	999	999	999	999	999
26	999	999	999	999	999	999	999	999	999	999	999	999
27	999	999	999	999	999	999	999	999	999	999	999	999
28	999	999	999	999	999	999	999	999	999	999	999	999
29	999	999	999	999	999	999	999	999	999	999	999	999
30	999	999	999	999	999	999	999	999	999	999	999	999
31	999	999	999	999	999	999	999	999	999	999	999	999
32	999	999	999	999	999	999	999	999	999	999	999	999
33	999	999	999	999	999	999	999	999	999	999	999	999
34	999	999	999	999	999	999	999	999	999	999	999	999
35	999	999	999	999	999	999	999	999	999	999	999	999
36	-0.1	999	999	999	999	999	999	999	999	999	999	999
37	999	999	999	999	999	999	999	999	999	999	999	999
38	999	999	-0.1	999	999	999	999	999	999	999	999	999
39	999	999	999	999	999	999	999	999	-0.2	999	999	999
40	999	999	999	999	999	-0.1	999	999	999	999	999	999
41	999	999	999	999	999	999	999	999	999	999	999	999
42	999	999	999	999	999	999	999	999	999	999	999	999
43	999	999	999	999	999	999	999	999	999	999	999	999
44	999	999	999	999	999	999	999	999	999	999	999	999
45	0	0	999	999	999	999	999	999	999	999	999	999
46	0	0	0	999	999	999	-0.1	999	999	999	999	999
47	999	0	0	999	999	999	999	-0.2	999	999	999	999
48	999	999	999	0	0	999	999	999	999	999	999	999
49	999	999	999	0	0	0	999	999	999	-0.1	999	999
50	999	999	999	999	0	0	999	999	999	999	-0.2	999
51	999	+0.1	999	999	999	999	0	999	999	999	999	999

52	999	999	+0.2	999	999	999	999	0	999	999	999	999
53	999	999	999	999	999	999	999	999	0	999	999	999
54	999	999	999	999	+0.1	999	999	999	999	0	999	999
55	999	999	999	999	999	+0.2	999	999	999	999	0	999
56	999	999	999	999	999	999	999	999	999	999	999	0